APPLE Memo

Nicole Bréaud - Pouliquen

GRUPPO
EDITORIALE
JACKSON

EDIZIONE
ITALIANA

APPLE Memo

Nicole Bréaud - Pouliquen

GRUPPO
EDITORIALE
JACKSON
Via Rosellini, 12
20124 Milano

© Copyright per l'edizione originale P.S.I. Editions du P.S.I. 1981
© Copyright per l'edizione Italiana Gruppo Editoriale Jackson - 1984

Il Gruppo Editoriale Jackson ringrazia per il prezioso lavoro svolto nella stesura dell'edizione italiana la signora Francesca Di Fiore, e l'Ing. Roberto Pancaldi.

Traduzione italiana a cura di Piero Dell'Orco

Tutti i diritti sono riservati. Stampato in Italia. Nessuna parte di questo libro può essere riprodotta, memorizzata in sistemi di archivio, o trasmessa in qualsiasi forma o mezzo, elettronico, meccanico, fotocopia, registrazione o altri senza la preventiva autorizzazione scritta dell'editore.

Fotocomposizione: Lineacomp S.r.l. - Via Rosellini, 12 - 20124 Milano

Stampato in Italia da:
S.p.A. Alberto Matarelli - Milano - Stabilimento Grafico

PRESENTAZIONE

Questo libro è destinato a risiedere, in permanenza, a fianco del vostro Apple quando lo utilizzate. Esso vi ricorda tutte le informazioni relative a riferimenti che potranno essere reperite velocemente: sintassi dei comandi, codici dei caratteri, messaggi di errore, linguaggio macchina, connessioni e indirizzi utili.

Le informazioni sono date senza eccessivi dettagli poichè lo scopo principale di questo libro è quello di fornire un rapido accesso alle informazioni; per una esposizione più introduttiva e completa, si potrà far riferimento ai libri "Alla scoperta dell'Apple II" e "Apple II - Guida all'uso".

Si termina con una raccolta di "trucchi" di differenti livelli, ma tutti utili, i "come...?" sono disposti senza un ordine particolare, è comunque possibile il loro rapido reperimento grazie ad un apposito indice.

Saremo riconoscenti a tutti i lettori per i suggerimenti e le informazioni supplementari che potranno essere inseriti in una prossima edizione.

SOMMARIO

CAPITOLI Pagina

I - **COMANDI**

Funzioni del BASIC Applesoft 1
Istruzioni del BASIC Applesoft 5
Operatori BASIC 11
Altri comandi 12
Applesoft 13
Funzioni e istruzioni dell'Integer BASIC 15
Funzioni Integer BASIC 16
Istruzioni Integer BASIC 18
Sistema operativo Pascal-UCSD 21
Editor Pascal-UCSD 23
Sistema operativo Pascal-UCSD: gestione dei file 26
I comandi del Monitor 28
Mini-assemblatore 31
I comandi del sistema operativo DOS 33

II - **CARATTERI**

Conversione esadecimale/decimale/esadecimale 40
I codici della tastiera 41
I codici dello schermo 42
ASCII (positivo) (bit 7 = 0) 45
ASCII (negativo) (bit 7 = 1) 46
Conversione esadecimale/decimale 47

III - **MESSAGGI DI ERRORE**

Applesoft 49
Messaggi di errore relativi ai file 56
Messaggi di errore relativi ai file DOS 57

IV - **IL LINGUAGGIO MACCHINA**

I registri interni del 6502 60
Il set di istruzioni del 6502 61

V - **I COME...?**

ICome...? 70

INDIRIZZI

Indirizzi del Monitor 86
Indirizzi del Monitor e della ROM Autostart 88
Indirizzi di sistema 89
Indirizzi di sistema con la Language Card 92
Indirizzi di sistema - schede di interfaccia 94
Indirizzi memoria ROM 95
Indirizzi del Monitor 96
Indirizzi fondamentali 103
Applesoft - puntatori fondamentali 105
Applesoft - esempio n.1 107
Applesoft - esempio n.2 110
Applesoft 112
Indirizzi dell'interprete Applesoft 115
Integer BASIC - puntatori fondamentali 122
Integer BASIC - esempio 123
Indirizzi Integer BASIC 129
DOS: indirizzi dischetto 131
Comandi DOS 133
DOS: indirizzi memoria RAM 134
Routine RWTS 135
DOS: indirizzi pagina 3 136
DOS: programmi di utilità 137
DOS - esempio 138

Una funzione chiede un argomento (oppure più argomenti) e restituisce un valore che è il risultato dell'applicazione di tale funzione al valore dell'argomento.

Funzioni matematiche

ABS Valore assoluto dell'argomento posto tra parentesi.

ATN Arcotangente - il risultato è dato in radianti, compreso tra -PI/2 e +PI/2.

COS Coseno - l'argomento dev'essere in radianti. Esempio: cos(x in gradi) => COS(PI*X/180).

EXP Esponenziale e^x. L'argomento dev'essere minore o uguale a 88, in caso contrario si produce un supero di capacità.

INT Parte intera, o meglio la più grande parte intera inferiore o uguale all'argomento: INT(0.5) vale -1; INT(-3) vale -3.

LOG Logaritmo naturale (neperiano o in base e). Per ottenere il logaritmo di X in base Y, utilizzare LOG(X)/LOG(Y). Esempio: logaritmo decimale di X => LOG(X)/LOG(10).

RND Fornisce, con un argomento positivo, un numero pseudo-casuale superiore o uguale a 0 e inferiore a 1. Esempio: PRINT RND(1) => .103112573. Se le chiamate alla funzione si succedono, i risultati costituiranno sempre la stessa serie di numeri pseudo-casuali. Una chiamata della funzione con un argomento negativo permette di uscire da una particolare serie. RND(0) restituisce l'ultimo numero generato.

SGN Funzione "segno": 1 se X>0, -1 se X<0 e 0 se X=0.

SIN Seno - l'argomento è supposto in radianti.

SQR Radice quadrata - l'argomento dev'essere superiore o uguale a 0.

TAN Tangente - l'argomento è supposto in radianti.

Funzioni di tabulazione

POS POS(0) fornisce la posizione libera successiva sulla linea dello schermo (posizione orizzontale del cursore).

SPC Questa funzione può essere impiegata solo nell'istruzione PRINT. PRINT SPC(X) stampa X spazi. X dev'essere un numero intero compreso tra 0 e 255.

TAB Questa funzione può essere impiegata solo nell'istruzione PRINT. PRINT TAB(X) sposta il cursore alla posizione di stampa X (1 è la posizione più a sinistra di una linea, 40 la più a destra) X dev'essere compreso tra 1 e 255 TAB(0) sposta il cursore alla posizione 256. Se X è minore dell'attuale posizione del cursore, non accade nulla. TAB non sposta mai il cursore verso sinistra.

HTAB Posiziona il cursore orizzontalmente prima dell'istruzione PRINT. HTAB1 corrisponde alla posizione più a sinistra sullo schermo. La posizione estrema vale 255 (5 linee più lontano).

VTAB Posiziona il cursore verticalmente prima dell'istruzione PRINT. VTAB1 corrisponde alla linea superiore sullo schermo. VTAB24 posiziona il cursore alla linea inferiore sullo schermo. Se l'argomento è superiore, rispetto alla linea più bassa sullo schermo, la visualizzazione potrà essere effettuata solo alla linea "puntata" dall'argomento per ogni successiva istruzione PRINT.

Funzioni di sistema

FRE Qualunque sia il valore dell'argomento, fornisce il numero di byte disponibili in memoria. Realizza anche la pulizia delle stringhe abbandonate.

PEEK Fornisce il contenuto (compreso tra 0 e 255) della locazione di memoria il cui indirizzo è uguale al suo argomento (che dev'essere un numero intero compreso tra 0 e 65535).

USR Chiamata di un programma utente in linguaggio macchina. L'unico argomento è trasmesso l'accumulmatore. L'indirizzo del sottoprogramma dev'essere preregistrato in $0B e $0C con JMP ($4C) in $0A. Il risultato è posto nell'accumulatore

Funzioni delle stringhe di caratteri

LEN(X$) Lunghezza (da 0 a 255).

LEFT$(X$,N) Estrazione degli N caratteri più a sinistra.

RIGHT$(X$,N) Estrazione degli N caratteri più a destra.

MID$(X$,K) oppure MID$(X$,K,N) estrazione fra tutti i caratteri o di N carateri a partire dalla Kiesima posizione. K dev'essere maggiore o uguale a 1.

Funzione di conversione

ASC(X$) Restituisce il codice ASCII del primo carattere della stringa X$. ASC("A") vale 65.

CHR$(K) Restituisce il carattere il cui codice ASCII vale K. CHR$(4) è CTRL-D.

STR$(A) Fornisce la rappresentazione di un numero in stringa di caratteri a partire dal suo valore numerico A.

VAL(X$) Fornisce il valore numerico rappresentato dalla stringa X$.

Funzioni grafiche (bassa risoluzione)

COLOR= Fornisce un colore (da 0 a 15) per un succesivo tratto in bassa risoluzione.

PLOT X,Y Disegna un piccolo quadrato all'ascissa X e all'ordinata Y. X e Y devono essere comprese tra 0 e 39. 0,0 è l'estrema posizione in alto a sinistra.

HLIN X1,X2 AT Y Traccia una linea orizzontale tra X1 e X2 all'ordinata Y.

VLIN Y1,Y2 AT X Traccia una linea verticale tra Y1 e Y2 all'ascissa X.

SCRN(X,Y) Restituisce il colore del quadratino tracciato in X,Y.

Funzioni grafiche (alta risoluzione)

HCOLOR= Fissa il colore (0, 1, 2, 3) del successivo punto da tracciare.

HPLOT X,Y Disegna un punto all'ascissa X e all'ordinata Y. X dev'essere compreso tra 0 e 279. Y dev'essere compreso tra 0 e 159 (HGR) oppure tra 0 e 191 (HGR2). 0,0 è l'estremo in alto a sinistra.

HPLOT X1,Y1 TO X2,Y2 Traccia una linea tra due punti, il comando può essere esteso ad altri punti ... TO Xn,Yn.

DRAW F AT X,Y Disegna la figura n. F della tabella delle figure (shapes) partendo dal punto X,Y.

XDRAW F AT X,Y Disegna la figura n. F della tabella delle figure (shapes). Il colore di ogni punto è il complemento dell'attuale colore del punto visualizzato sullo schermo.

ROT= L'argomento è proporzionale all'angolo di rotazione che si desidera dare alla figura da disegnare con la funzione DRAW. ROT=16 corrisponde ad una rotazione di 90°.

SCALE= Fornisce un valore d'ingrandimento alla figura da disegnare compreso tra 1 e 255.

Funzioni relative ai comandi per i giochi (paddle)

PDL(X) Restituisce un numero da 0 a 255 proporzionale alla posizione angolare del potenziometro di cui è dotato il comando (paddle). X vale 0, 1, 2 oppure 3.

PEEK(X-16287) Restituisce un risultato maggiore di 127 se il pulsante sul comando X è stato premuto. X vale 0, 1 oppure 2.

Funzione altoparlante

PEEK(-16336) Emette un "clik" dall'altoparlante.

Modo obbligatorio	*Parola chiave*	*Definizione - Esempio*
	&	Rimanda l'esecuzione di un sottoprogramma in linguaggio macchina il cui indirizzo si trova alla locazione $3F6; $3F7 con JMP($4C) alla locazione $3F5 &16 → 10 se il sottoprogramma è convertito da decimale a esadecimale.
	CALL	Rimanda l'esecuzione di un sottoprogramma in linguaggio macchina all'indirizzo specificato. CALL-151 Un argomento negativo equivale al complemento di 65536 dell'indirizzo cercato.
	CLEAR	Rimette a zero tutte le variabili. Le stringhe hanno tutte lunghezza nulla.
Diretto	**CONT**	Prosegue l'esecuzione di un programma interrotto.
Diretto	**CTRL-C**	Arresta l'esecuzione di un programma in corso. Il programma resta intatto.
	CTRL-D	Inizia un comando DOS dev'essere preceduto da PRINT".
Diretto	**CTRL-S**	Sospende la visualizzazione, l'immagine resta fissa fino a quando non viene premuto un tasto qualsiasi.
Diretto	**CTRL-X**	Annulla le impostazioni di una linea oppure un dato appena impostato.
	DATA	Definisce un elenco di costanti che saranno lette dall'istruzione READ 10 DATA ABC,5,0.15
	DEL	Con due argomenti separati da una virgola, delimita una parte di programma da cancellare DEL 10,50 sopprime le istruzioni da 10 a 50

Modo obbligatorio	*Parola chiave*	*Definizione - Esempio*
	DEF FN	Definisce una funzione dell'utente con un solo argomento: 10 DEF FN F(X)=X-256*INT(X/256)
	DIM	Dimensionamento di una matrice (fissa il numero e i valori massimi degli indici. *10 DIM A(100),B%(500),C$(10)* *20 DIM T(N)* *30 DIM M(10,10,10)* 88 è il numero massimo di indici
Programmato	**END**	Arresta l'esecuzione relativa alla serie di istruzioni.
	FOR	Introduce un ciclio (loop): tutte le istruzioni comprese tra *FOR I=A TO B STEP C* e NEXT I saranno ripetute per tutti i valori di I compresi tra A e B, con passo C *10 FOR I=1 TO 1000* *20 FOR X=1.5 TO 2 STEP .1* *30 FOR J=N TO -N STEP -2* Se più cicli si succedono con il medesimo indice, non è possibile interrompere la progressione dell'indice fino al suo valore massimo *10 FOR I=1 TO 100* *20 IF N$(I)=X$ THEN T=I:I=100* *30 NEXT I* *40 IF T=0 THEN PRINT"NON TROVATO":END* *50 PRINT"TROVATO IN";T*
	FLASH	Visualizza i caratteri in modo lampeggiante. Questo modo non può essere neutralizzato da RESET. Il tasto → modifica i caratteri sullo schermo. Battere NORMAL per ristabilire la situazione.

Modo obbligatorio	*Parola chiave*	*Definizione - Esempio*
Programmato	**GET**	Attende un carattere da testiera, che poi non viene visualizzato. CTRL-C non ha alcun effetto *10 GET A$* non è raccomandato con istruzioni del DOS nel programma, salvo che CTRL-D sia preceduto da un RETURN *D$=CHR$(13)+CHR$(4)*
	GOSUB	Chiamata ad un sottoprogramma 10 GOSUB 1000
	GOTO	Salta ad un'altra istruzione numerata 10 GO TO 50
	GR	Pone una parte dello schermo in visualizzazione grafica da 40x40 quadratini. Lascia 4 linee di testo in basso.
	HGR	Pone una parte dello schermo in visualizzazione grafica da 280x160 punti. Lascia 4 linee di testo.
	HGR2	Pone tutto lo schermo in visualizzazione grafica da 280x192. Il cursore non viene visualizzato.
	HIMEM:	Specifica la più alta posizione di memoria RAM utilizzabile dal programma.
	HOME	Pulisce lo schermo e posiziona il cursore in alto a sinistra. Preceduto da TEXT: pulisce tutto lo schermo.
	IF	Salto condizionato, nella forma IF <condizione> THEN <istruzione>. Se la condizione non è soddisfatta (risultato falso o uguale a 0) si passa alla linea seguente; se la condizione è soddisfatta viene eseguita l'istruzione che segue THEN IF C THEN GOTO x oppure

Modo obbligatorio	*Parola chiave*	*Definizione - Esempio*
		IF C THEN x oppure ancora IF C GOTO x *10 IF A>B THEN Y=K* *20 IF A$=""THEN 5* *30 IF A<0 OR A>100 THEN 100*
Programmato	**INPUT**	Acquisizione di un dato da tastiera *10 INPUT A* *20 INPUT A,B,C$,D* *30 INPUT "IL VOSTRO NOME ?";N$*
	IN#	Connette in entrata la periferica collegata al connettore indicato nell'argomento.
	INVERSE	Provoca la visualizzione dei caratteri in nero su bianco. Per ritornare alla consueta visualizzazione battere NORMAL.
	LET	Istruzione di assegnazione di un valore ad una variabile. Non è obbligatoria *LET X$="AGOSTO"*
	LIST	Lista un programma *LIST tutto il programma* *LIST 10,100 da 10 a 100* *LIST 100, da 100 alla fine* *LIST ,10 fino a 10* La virgola può essere sostituita con -
	LOAD	Carica un programma da cassetta in memoria RAM.
	LOMEM:	Specifica la posizione più bassa di memoria in RAM disponibile per le variabili del programma.
Diretto	**NEW**	Cancella il programma attualmente in memoria RAM. (2 puntatori sono posti a zero).
	NEXT	Provvede alla successiva iterazione del ciclo FOR *NEXT I NEXT J,I NEXT*

Modo obbligatorio	*Parola chiave*	*Definizione - Esempio*
	NORMAL	Ripristina sullo schermo la visualizzazione in bianco su nero.
	NOTRACE	Disabilita il modi TRACE
	ON	ON I GOTO 10,20,30 Se I vale 1, si va alla linea 10, se vale 2 si va alla linea 20, alla linea 30 se vale 3. Se I è nullo o falso, si passa all'istruzione successiva. ON I GOSUB 1000,3000 Se I vale 1, viene chiamato il sottoprogramma alla linea 1000, se vale 2 quello della linea 3000.
	ONERR	ONERR GOTO 500 Consente di intercettare un errore prima che questo provochi l'arresto dell'esecuzione del programma. Quando è riscontrato un errore il programma salta alla linea indicata.
	POKE	POKE a,b scrive il dato "b" all'indirizzo assoluto "a" ("a" e "b" sono espressi in decimale. *POKE 2000,65*
	POP	Elimina dallo stack l'ultimo indirizzo di ritorno di un sottoprogramma. Il RETURN successivo porta all'istruzione che segue l'ultimo GOSUB eseguito.
	PRINT	Visualizza un risultato sullo schermo o sulla stampante PRINT A *10 PRINT A;B;J (uno accanto all'altro)* *30 PRINT "X=";X* *20 PRINT A,B,J (in zone fisse)*
	PR#	Trasferisce l'uscita dei dati alla periferica collegata alla scheda nel connettore specificato nell'argomento *PR#1* consente l'uscita sulla stampante se questa è collegata allo slot 1.

Modo obbligatorio	*Parola chiave*	*Definizione - Esempio*
Programmato	**READ**	Lettura dei dati contenuti nell'istruzione DATA associata *10 READ A* *20 READ B$,C*
	RECALL	Recupero dei dati numerici dalla cassetta verso la memoria RAM. L'argomento è costituito da una variabile specifica e correttamente dimensionata. *5 DIM B(100)* *100 RECALL B*
Programmato	**REM**	Introduce un commento nel listato di un programma.
	RESET	Questo tasto equivale a CTRL-C durante l'esecuzione di un programma. Il programma si arresta ma rimane intatto. Le periferiche in linea sono disattivate. Il programma torna all'interprete oppure all'indirizzo previsto in $3F2, $3F3 se PWERDUP è conforme, altrimenti il sistema riparte cone all'accensione (COLDSTART).
	RESTORE	Torna all'inizio dei DATA
	RETURN	Ritorno da un sottoprogramma *100 RETURN*
	RESUME	Torna all'istruzione dove è stato riscontrato un errore trattato dal programma l'istruzione ONERR GOTO
	RUN	Ordina l'esecuzione di un programma. Rimette a zero tutte le variabili. *RUN RUN 30*
	SAVE	Salvataggio di un programma su cassetta.
	SPEED=	Modifica la velocità di visualizzazione sullo schermo da 1 (minimo) a 255 (standard).

Modo obbligatorio	*Parola chiave*	*Definizione - Esempio*
	STEP	Introduce il passo di incremento nel ciclo FOR NEXT.
	STOP	Arresta l'esecuzione di un programma. *10 STOP* visualizza il messaggio *?BREAK IN 10* Si può proseguire con il comando CONT (se le istruzioni non sono state modificate).
	STORE	Salvataggio di una matrice numerica su cassetta. Non funziona direttamente con le stringhe di caratteri *STORE A*
	TEXT	Riporta la visualizzazione in modo testo dopo la visualizzazione in modo grafico. Ripristina i valori standard relativi alla visualizzazione: 40 caratteri per linea 24 linee per ogni "videata"
	THEN	Introduce l'istruzione da effettuare quando un IF è soddisfatto.
	TO	Introduce il valore limite nel ciclo FOR.
	TRACE	Modo di verifica e correzione di eventuali errori. Visualizza il numero di istruzioni eseguite senza RETURN, quindi tra le linee dei risultati del programma.
	WAIT	Pausa condizionata in un programma. *WAIT A,B* Sospende l'esecuzione di un programma fino a quando ciò che è contenuto all'indirizzo A e (bit per bit) l'equivalente di B sia differente da 0 *WAIT -16384,128* corrisponde all'attesa della pressione di un tasto dalla tastiera.

OPERATORI BASIC

+	Addizione di numeri oppure concatenazione di stringhe di caratteri
-	Sottrazione
*	Moltiplicazione
/	Divisione
^	Elevazione a potenza
=	Uguale <> diverso
<	Minore > maggiore
<=	Minore o uguale
=<	Uguale o minore
>=	Maggiore o uguale
=>	Uguale o maggiore
NOT	Funzione logica NOT, agisce su un solo operando. Se A è vero NOT A è falso Se A è falso NOT A è vero
AND	Funzione logica AND su due operandi P AND Q non è vero solo se P e Q sono veri.
OR	Funzione logica OR su due operandi P OR Q non sono falsi solo se P o Q sono falsi.

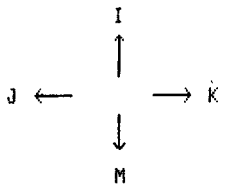
- *Posizionamento del cursore*

- *Visualizzazione di linee sullo schermo*

- *Copia di una linea in memoria RAM*

- *Soppressione di una linea dalla memoria RAM*

Con il Monitor AUTOSTART ROM in memoria ROM: i comandi di posizionamento del cursore sono ottenuti con i seguenti quattro tasti, preceduti dalla pressione del tastO ESC.



Utilizzando solo questi quattro tasti si resta in modo controllo cursore nelle quattro direzioni.

Per tornare al modo normale di inserimento e correzione, basta premere il tasto ESC.

Per cancellare, partendo dalla posizione del cursore, fino in fondo alla pagina premere ESC E.

Per cancellare, partendo dalla posizione del cursore, fino in fondo alla paginas premere ESC F.

Per cancellare tutto il contenuto dello schermo e posizionare il cursore nell'angolo in alto a sinistra, ESC (SHIF/P).

Usare il tasto ← (freccia sinistra) per annullare l'ultimo carattere inserito.

Usare il tasto → (freccia destra) per reinserire in memoria RAM i caratteri posti sotto il cursore. Per reinserire una serie di caratteri utilizzare il tasto → insieme al tasto REPT, si procederà più velocemente.

Per sopprimere una linea d'istruzione dal programma in memoria RAM, battere il numero di linea seguito da RETURN.

Le parole chiave in ordine alfabetico e i relativi codici esadecimali.

Ordine alfabetico dalla A alla F

Parola chiave	*Codice esad.*	*Parola chiave*	*Codice esad.*	*Parola chiave*	*Codice esad.*
&	$AF	GR	$88	ON	$84
ABS	$D4	HCOLOR=	$92	ONERR	$A5
AND	$CD	HGR	$91	OR	$CE
ASC	$E6	HIMEM:	$A3	PDL	$D8
AT	$C5	HLIN	$8E	PEEK	$E2
ATN	$E1	HOME	$97	PLOT	$8D
CALL	$8C	HPLOT	$93	POKE	$B9
CHR$	$E7	HTAB	$96	POP	$A1
CLEAR	$BD	IF	$AD	POS	$D9
COLOR	$A0	IN#	$8B	PRINT	$BA
CONT	$BB	INPUT	$84	PR#	$8A
COS	$DE	INT	$D3	READ	$87
DATA	$83	INVERSE	$9E	RECALL	$A7
DEF	$B8	LEFT$	$E8	REM	$B2
DEL	$85	LEN	$E3	RESTORE	$AE
DIM	$86	LET	$AA	RESUME	$A6
DRAW	$94	LIST	$BC	RETURN	$B1
END	$80	LOAD	$B6	RIGHT$	$E9
EXP	$DD	LOG	$DC	RND	$DB
FLASH	$9F	LOMEM:	$A4	ROT	$98
FN	$C2	MID$	$EA	RUN	$AC

Le parole chiave in ordine alfabetico e i relativi codici esadecimali.
(Segue)
Ordine alfabetico dalla F alla X

Parola chiave	*Codice esad.*	*Parola chiave*	*Codice esad.*	*Parola chiave*	*Codice esad.*
FOR	$81	NEW	$BF	SAVE	$B7
FRE	$D6	NEXT	$82	SCALE	$99
GET	$BE	NORMAL	$9D	SCRN(	$D7
GOSUB	$B0	NOT	$C6	SGN	$D2
GOTO	$AB	NOTRACE	$9C	SHLOAD	$9A
SIN	$DF	STR$	$E4	USR	$D5
SPC(	$C3	TAB(	$C0	VAL	$E5
SPEED=	$A9	TAN	$E0	VLIN	$8F
SQR	$DA	TEXT	$89	VTAB	$A2
STEP	$C7	THEN	$C4	WAIT	$B5
STOP	$83	TO	$C1	XPLOT	*
STORE	$A8	TRACE	$9B	XDRAW	$95

* XPLOT è codificato in: $58 $8D
"X" "PLOT"

Non può essere utilizzato come nome di variabile.

FUNZIONI E ISTRUZIONI DELL'INTEGER BASIC

Le parole chiave in ordine alfabetico dalla A alla V

Parola chiave	*Codice esad.*	*Parola chiave*	*Codice esad.*	*Parola chiave*	*Codice esad.*
ABS	$31	INPUT	$52 $53 $54	PRINT	$61 $62 $63
AND	$1D				
ASC	$3C	LEN	$3B	PR#	$7E
AT	$6B $6E	LET	$5E	REM	$5D
AUTO	$0D	LIST	$74 $75 $76	RETURN	$5B
CALL	$4D	LOAD	$04	RND	$2F
CLR	$0C	LOMEM	$11	SAVE	$05
COLOR=	$66	MAN	$0F	SCRN(	$3D
CON	$06	MOD		SGN	$30
DEL	$09	NEW	$0B	STEP	$58
DIM	$4E	NEXT	$59	TAB	$50
DSP	$7B $7C	NOT	$37	TEXT	$4B
END	$51	NOTRACE	$7A	THEN	$24
FOR	$55	NODSP	$78 $79	TO	$57
GOSUB	$5C	OR	$1E	TRACE	$7D
GOTO	$5F	PDL	$32	VLIN	$6C
GR	$4C	PEEK	$2E	VTAB	$6F
HLIN	$69	PLOT	$67		
IF	$60	POKE	$64		
IN#	$7F	POP	$77		

Funzioni matematiche

ABS	Valore assoluto.
MOD	Restituisce il resto della divisione tra il primo e il secondo operando. PRINT 15 MOD 4 restituisce il numero 3
RND	Genera un numero intero pseudo-casuale positivo e inferiore all'argomento PRINT RND(10) fornisce il numero 3

Funzioni di tabulazione

TAB	Equivale al comando HTAB dell'Applesoft. Posiziona il cursore, nello schermo, in una posizione tra 1 e 255.
VTAB	Posiziona verticalmente il cursore (in assoluto) tra 1 e 24.
VLIN Y1,Y2 AT X	Traccia una linea verticale tra Y1 e Y2

Funzioni di sistema

PEEK	Legge il contenuto di una posizione di memoria all'indirizzo indicato dall'argomento. L'argomento è compreso tra -32768 e +32767.

Funzioni di conversione

ASC("Z")	Restituisce il codice ASCII del carattere.

Funzioni grafiche

COLOR=	Fornisce un colore (da 0 a 15) per il succesivo tracciamento in bassa risoluzione.
PLOT X,Y	Posizione un piccolo quadrato nella posizione dell'ascissa X e della coordinata Y. X e Y devono essere comprese tra 0 e 39.0,0 corrisponde alla posizione in alto a sinistra.
HLIN X1,X2 AT Y	Traccia una linea orizzontale tra X1 e X2 alla coordinata Y.

FUNZIONI DELL'INTEGER BASIC

VLIN Y1,Y2 AT X	Traccia una linea verticale tra Y1 e Y2 all'ascissa X.
SCRN(X,Y)	Restituisce il colore del quadratino tracciato alla posizione X,Y.

Funzioni relative ai comandi dei giochi (Paddle)

PDL(X)	Restituisce un numero tra 0 e 255 proporzionale alla posizione angolare del potenziometro. X vale 0,1 oppure 2.
PEEK(X-16287)	Restituisce un valore maggiore di 127 se è premuto il tasto della paddle X. X vale 0,1 oppure 2.

Funzioni dell'altoparlante

PEEK(-16336)	Produce l'emissione di un 'clik' dall'altoparlante.

Parola chiave	*Definizione - Esempio*
AUTO	Numerazione automatica delle linee a partire dal numero indicato nell'argomento. AUTO 100,5: numerazione di 5 in 5 partendo dalla linea 100. Se il secondo argomento non è specificato la numerazione procede di 10 in 10.
CALL	Esegue un sottoprogramma che comincia all'indirizzo indicato (minore di 32767). CALL-939 pulisce lo schermo.
CLR	Rimette a zero tutte le variabili.
CON	Prosegue il programma dopo una interruzione.
CTRL-C	Arresta un programma.
DEL	Sopprime le linee di istruzioni *DEL 10,100*
DIM	- Dimensionamento di una matrice ad una sola dimensione. - Dimensionamento della lunghezza massima di una variabile stringa di caratteri *10 DIM N$(20)* (obbligatoria per tutte le variabili stringhe di caratteri).
DSP	Visualizza i nuovi valori assunti dalla variabile specificata durante l'esecuzione di un programma *10 X=RND(10)* *15 DSP X* *20 GOTO 10* *RUN → # 10 X=3* *# 10 X=7 etc.*
END	Ultima istruzione di un programma BASIC. Se manca tale istruzione viene prodotto un messaggio di errore.
FOR	Ciclo FOR-NEXT che ripete, per ogni ciclo, le istruzioni comprese tra FOR e NEXT, oltre al conteggio e alla verifica della fine ciclo.
GOSUB	Chiamata di un sottoprogramma ad un indirizzo che può essere rappresentato da una espressione aritmetica oppure da una variabile *GOSUB 3*X+10*

Parola chiave	*Definizione - Esempio*
GOTO	Salta ad una linea di istruzione il cui numero può essere calcolato nel programma *GOTO N*10*
GR	Modo grafico con risoluzione 40*40
IF	Salto condizionale nella forma: IF condizione THEN istruzione V : istruzione F. Se la condizione non è soddisfatta l'istruzione V è ignorata, si passa quindi all'istruzione F. Se la condizione è soddisfatta l'istruzione V viene eseguita prima dell'istruzione F.
INPUT	Acquisizione di un dato da tastiera. E' necessario separare il messaggio dalla variabile con una virgola. *10 INPUT "QUAL'E' IL TUO NOME ? ",N$* Durante l'esecuzione viene visualizzato un punto di domanda davanti al cursore se non è previsto un messaggio oppure se la variabile è numerica.
IN#	Riceve i dati da una periferica. L'argomento è il numero del connettore (da 1 a 7) corrispondente al collegamento della periferica.
LET	Assegna un valore ad una variabile.
LOAD	Caricamento di un programma da cassetta in memoria RAM.
LOMEM:	Modifica in memoria il posizionamento delle variabili.
MAN	Termina il modo automatico di numerazione delle linee. Quando appare un nuovo numero di linea spostare il cursore fino a > quindi battere *MAN*.
NEW	Cancella il programma in memoria RAM.
NEXT	Comanda il passaggio al successivo ciclo FOR.
NOTRACE	Termina la funzione TRACE.
NODSP	Termina la funzione di visualizzazione dei cambiamenti relativi ai valori delle variabili chiesti con l'istruzione DSP.

Parola chiave	*Definizione - Esempio*
POKE	POKE a,b scrive il dato "b" (inferiore a 256) nella cella di memoria che ha l'indirizzo "a" *(da -32768 a 32767)*.
POP	Elimina un livello di "return" da un sottoprogramma.
PRINT	Visualizza un risultato sullo schermo.
PR#	Seleziona la periferica di uscita.
REM	Commento in una linea di programma.
RETURN	Comanda il ritorno da un sottoprogramma.
RUN	Esegue un programma, le variabili dimensionate non sono rimesse a zero.
SAVE	Salva un programma su cassetta.
STEP	Definisce il passo di incremento nel ciclo FOR.
TEXT	Riporta il calcolatore al modo testo dopo l'utilizzo in modo grafico. Ripristina il modo di visualizzazione standard.
THEN	Definisce l'istruzione da effettuare quando un IF è soddisfatto. La **seconda** istruzione dopo THEN sarà effettuata se la condizione è falsa.
TRACE	Se più istruzioni sono sulla medesima linea, la funzione TRACE indicherà solo il passaggio relativo alla prima istruzione della linea.

I nomi delle variabili in Integer BASIC sono conservati interamente quale che sia la loro lunghezza.

In Integer BASIC sono possibili solo operazioni intere.

Gli errori di sintassi sono riconosciuti subito dopo aver convalidato la linea con RETURN.

Operatori

+ - * /	(divisione intera)
^	(evelazione a potenza)
=	(uguale)
= >,< #	(diverso da)
NOT, AND, OR	

SISTEMA UCSD - PASCAL

Configurazione standard

+ 48 Kbyte di memoria RAM
+ Language Card, con 16 Kbyte di memoria RAM, sullo slot 0
+ 2 unità per floppy disk Disk II
+ 1 controller per floppy disk con PROM P5A e P6A (16 settori) nello slot 6.

I dischetti del sistema Pascal

APPLE 1: } Sistema } Editor, Filer, Apple Library
APPLE 2: Compilatore, Linker, Assembler, etc.
APPLE 3: Dimostrazione capacità grafiche, Formatter, Library
BASIC S: Per il passaggio in BASIC (sotto DOS 3.2) e caricamento di un interprete nella Language Card.

Formattazione dei dischetti vergini sotto Pascal:

X (esecuzione) del programma APPLE 3 : FORMATTER
Porre il dischetto vergine in D2 (Drive 2)
Rispondere 5 alla domanda FORMAT WHICH DISK (4,5,9...12) ? oppure RETURN per rinunciare alla formattazione.

Copia dei dischetti:

F (gestione dei dischetti) seguito da T(ransfert)
TRANSFERT ? nome del dischetto da copiare:
TO WHERE ? BLANK :
TRANSFERT 280 BLOCKS ? Y
DESTROY BLANK : ? Y

Comandi

E(dit	Chiama l'editor.
R(un	Chiama un programma sorgente (.TEXT), lo compila e lo esegue.
F(iler	Gestione degli archivi e dei programmi su dischetto.
C(omp	Compilazione.
L(ink	Unisce due programmi già compilati.

X(ecute	Esecuzione di un programma oggetto (.CODE).
A(ssem	Chiama l'assembler.
D(ebug	Non è utilizzato.
H	Rinizializza il sistema con un dischetto DOS 3.3 o con il dischetto BASICS (per il DOS 3.2).
CTRL-A	Visualizzazione dei caratteri dalla colonna 41 alla colonna 80 (parte destra dello schermo).
CTRL-K	Per l'impostazione del carattere [.
SHIFT-M	Per l'impostazione del carattere].

>EDIT A(DJUST C(OPY D(ELETE F(IND I(NSERT
J(UMP R(EPLACE Q(UIT X(CHANGE Z(AP S(ET
V(ERIFY

Direzione di posizionamento del cursore nel testo

> normale con → e ←; CTRL-A visualizza la parte destra
< indietro CTRL-L posiziona verso il basso
CTRL-O posiziona verso l'alto
CTRL-Q posiziona ad inizio linea
CTRL-Z visualizza seguendo il cursore

Modifica della direzione del cursore

- oppure , per <
+ oppure . per >
<BS> un carattere indietro con la freccia sinistra ←
<EXT> fine delle operazioni con CTRL-C
<ESC> annullamento di operazione con il tasto ESC
<DEL> annullamento di una linea di testo con CTRL-X

Impaginazione >A(DJUST : L(JUST R(JUST C(ENTER
<LEFT, RIGHT, UP, DOWN ARROWS> [<EXT> TO LEAVE]

L	posizionamento del testo sul margine sinistro	
R	allineamento del testo sul margine destro	
C	centratura del testo nella pagina	
LEFT	freccia ←	posizionamento del testo a sinistra
RIGHT	freccia →	posizionamento del testo a destra
UP	CTRL-O	allineamento della linea precedente
DOWN	CTRL-L	allineamento della linea seguente
<EXT>	CTRL-C	per lasciare la funzionze

Copia >C(OPY : B(UFFER F(ROM FILE <ESC>

B copia tampone (per esempio di un testo che sarà visualizzato).
F copia di un file dal dischetto dove si trova il cursore.
<ESC> il tasto ESC consente di lasciare la funzione

Cancellazione >D(ELETE : <>MOVING COMMANDS [<EXT> TO DELETE,
<ESC> TO ABORT]

	freccia ←	per cancellare un carattere
	freccia →	per riprendere il testo cancellato
<EXT>	CTRL-C	per convalidare la cancellazione
<ESC>	ESC	per annullare la cancellazione

Verifica >V(ERIFY Verifica dello schermo dopo le modifiche

Ricerca >F(ind [1] : L(IT <TARGET> ⇒

La frase da cercare è compresa tra / e /. L(ITTERAL se il testo è incluso nella frase. I n I alla ennesima occorrenza della frase. Questo numero dev'essere indicato prima di battere F.

Inserzione >I(NSERT TEXT [<BS> A CHAR, <DEL> A LINE]
[<EXT> TO ACCEPTS, <ESC> ESCAPES]

Il testo da inserire è normalmente inserito correggendo con la freccia <- oppure <BS> per il carattere precedente o, ancora, CTRL-X per annullare una linea. <EXT> oppure CTRL-C per convalidare l'inserimento. <ESC> per annullare l'operazione.

Salto >J(UMP : B(EGINNING E(ND M(AKER <ESC>

B	Per posizionare il cursore all'inizio del testo.
E	Per posizionare il cursore alla fine del testo.
M	Per posizionare il cursore ad un marker prefissato (vedere S(ET e M(ARKER).
<ESC>	Per lasciare la funzione.

Uscita dall'editor >Q(UIT : U(PDATE THE WORKFILE AND LEAVE
E(XIT WITHOUT UPDATING
R(ETURN TO THE EDITOR WITHOUT UPDATING
W(RITE TO A FILE NAME AND RETURN

U	Aggiorna il file di lavoro chiamato SYSTEM.WRK.TEXT.
E	Uscita senza aggiornamento del file.
R	Ritorno all'editor senza salvare il file.
W	Salvataggio del file specificato e ritorno all'editor.

Margini >S(ET : E(NVIROMENT M(ARKER <ESC>
E(NVIROMENT : [OPTIONS] <ESC> OR <SP> TO LEAVE

A(UTO INDENT TRUE indentazione automatica (allineamento sulla linea precedente).

F(ILLING	FALSE	riempimento fino al margine destro.
L(EFT MARGIN	0	margine sinistro.
R(IGHT MARGIN	78	margine destro.
P(ARA MARGIN	5	margine relativo al paragrafo.
C(OMMAND CH	^	carattere di comando nel modo M(ARGIN
T(OKEN DEF	TRUE	presenta il modo F(IND e R(EPLACE.

Nell'edizione di un programma scritto in Pascal, A deve restare TRUE e F dev'essere FALSE.
<SP> barra-spazio per uscire dalla funzione.

Sostituzione >R(EPLACE I n I : L(IT V(FY <TARGET> <SUB> ⇒

Sostituzione del testo compreso tra / e / con un nuovo testo compreso tra / e / di lunghezza qualsiasi.

L Per sostituire una parte di testo.
n Numero delle operazioni di sostituzione da effettuare (dev'essere impostato prima del comando R).

Cambio di caratteri >E X(CHANGE : TEXT I <BS> A CHAR I
I <ESC> ESCAPES, <EXT> ACCEPTS I

Il carattere sostituisce quello posto sotto il cursore.
<BS> freccia ← per il carattere precedente.
<ESC> Tasco ESC per annullare.
<ETX> CTRL-C per convalidare.

Cancellazione >Z(AP cancellazione dopo la corrente posizione cursore fino a quella d'inizio dell'ultimo testo trovato, sostituisce o inserisce.

F(ILER G(ET S(AVE N(EW L(IST DIRECTORY
E(XTENDED-DIRECTORY LIST R(EMOVE C(HANGE
T(RANSFERT D(ATE Q(UIT V(OLUME W(HAT
B(AD-BLOCKS X(AMINE Z(ERO P(REFIX

G GET ? Nome del dischetto : **Nome di programma**
Caricamento in memoria del programma specificato, sostituisce il SYSTEM.WRK.TEXT (file di lavoro).
TEXT FILE LOADED, l'operazione è stata realizzata.

S SAVE AS ? **Nome del dischetto : Nome di programma**
Salvataggio di un file di lavoro sotto il nome specificato nel dischetto indicato.
TEXT FILE SAVED, l'operazione è stata realizzata.

N Cancellazione di un file di lavoro in memoria RAM e nel dischetto. Il calcolatore risponde : WORKFILE CLEARED.

L DIR LISTING OF ? **Nome del dischetto :**
Visualizza il contenuto del dischetto (catalog).
Far seguire il nome da ,PRINTER : per ottenere una stampa su carta.

E Visualizza del contenuto con informazioni diverse come quelle presenti nelle zone inutilizzate.
DIR LISTING OF ? **Nome del dischetto :**

R Soppressione di un file.

C Modifica il nome di un file sul dischetto.

T Trasferimento di un dischetto o di un file su un altro dischetto.
TRANSFER ? **Nome del dischetto : [Nome del programma]**
TO WHERE ? **Nome del dischetto : [Nome del programma]**
Per stampare un programma sorgente rispondere PRINTER: alla domanda TO WHERE ?.

D Aggiornamento della data.
Giorno - Mese (3 lettere) - Anno (2 cifre).

Q Permette di lasciare le funzioni del Filer.

V Elenco dei volumi conosciuti dal sistema ordinati per numero e per nome.

W WHAT fornisce i nomi dei file di lavoro e indica se sono stati salvati oppure no.

B Bad-blocks verifica i 280 blocchi di un dischetto e segnala i blocchi in cattivo stato.

X Esame dei blocchi che sono risultati in cattivo stato e tentativo di ripristino. Se ciò non risulta possibile consente di marcare i blocchi danneggiati (operazione utile prima dell'utilizzazione di un dischetto vergine).

Z Zero, cancella la DIRECTORY (elenco degli archivi).

P Prefisso, consente la modifica del nome relativo al volume corrente assunto per default battendo soltanto ":".

Il "promp" (pronto) è rappresentato dal segno *.
I dati sono forniti in notazione esadecimale.
Gli indirizzi sono forniti con 4 cifre esadecimali.

Comando	*Definizione - esempio*
Indirizzo G	Esecuzione di un programma che inizia a questo indirizzo. *3D0 G : rinizializzazione, 'a caldo' del BASIC.*
Indirizzo L	Elenca 20 istruzioni in linguaggio macchina partendo da questo indirizzo; disassembla i codici esadecimali nei codici mnemonici del Mini-Assembler. *3D0 L* *3D0 - 4C BF 9D JMP $9DBF* *3D3 - 4C 84 9D JMP $9D84* *... etc.*
Indirizzo 1.Indirizzo 2	Visualizza il contenuto delle posizioni di memoria partendo dall'indirizzo 1 fino all'indirizzo 2. *3D0.3D7* *3D0 - 4C BF 9D 4C 84 9D 4C FD* Gli indirizzi di inizio linea sono sempre nella forma XXX0 oppure XXX8 salvo eventualmente l'indirizzo 1.
Indirizzo	Se viene specificato un solo indirizzo il Monitor rinvia il contenuto della cella di memoria specificata. *3D1* *3D1 - BF*
Indirizzo:valore 'spazio' valore	Modifica o scrittura dei valori in posizioni di memoria adiacenti partendo dall'indirizzo specificato. *73: 00 20 (modifica HIMEM).*

Comando	*Definizione - esempio*
Indirizzo 1 < Indirizzo 2. Indirizzo 3 M	Spostamento di unà zona di valori contenuti partendo dall'indirizzo 2 fino all'indirizzo 3, nella zona che comincia all'indirizzo 1. *6000<400.7FF M* salvataggio della pagina di testo o grafica nella zona di memoria compresa tra $6000 e $67FF.
Indirizzo 1 < Indirizzo 2. Indirizzo 3 V	Verifica l'identità tra 2 zone di memoria. Eventuali differenze sono segnalate: *F10B<B1.C8 V* *00B8 - 05 (60)* *00B9 - 02 (EA)* Il sottoprogramma CHRGET nella sua forma originale differisce, nella forma attuale per i contenuti di $B8 e $B9 (puntatore nel testo BASIC).
N	Visualizzazione in modo normale e separatore tra successivi comandi MONITOR.
I	Visualizzazione in modo inverso (nero su bianco).
Valore + Valore	Operazioni di addizione e sottrazione nel sistema di numerazione esadecimale (2 cifre). *3F+01 oppure 40-01* *40 3F*
Numero di connettore CTRL-K	Trasferisce una entrata in arrivo dalla periferica collegata allo slot specificato. Trasferisce il controllo dell'uscita alla periferica collegata allo slot specificato. 6 *CTRL-P* attiva la PROM della scheda di controllo dei drive per floppy disk provocando l'inizializzazione del sistema.

Comando	*Definizione - esempio*
CTRL-B	Caricamento dell'interprete BASIC nella memoria RAM (inizializzazione 'a freddo').
CTRL-C	Ricarica dell'interprete BASIC nella memoria RAM (uguale a RESET)
CTRL-E	Visualizzazione del contenuto dei registri del microprocessore. *A= X= Y= P= S=* Modifica di tali registri.
CTRL-Y	Salta al programma che comincia all'indirizzo. *$3F8* *3F8 : 4C 00 03 JMP $300* $3F8 dev'essere preparato per provocare il salto all'inizio del programma chiamato.
Indirizzo 1. Indirizzo 2 W	Registra su cassetta la zona di memoria posta tra l'indirizzo 1 e l'indirizzo 2 **4000.5FFF W*
Indirizzo 1. Indirizzo 2 R	Lettura da cassette. Carica in memoria RAM, partendo dall'indirizzo 1 fino all'indirizzo 2, dati e programmi registrati sulla cassetta.

Comando	*Definizione - esempio*
* F666 G oppure CALL-2458	Entrata nel Mini-Assembler la risposta è costituita da "!".
adrs : COP spazio OPE	Istruzione assembler all'indirizzo 'adrs' (4 cifre esadecimali); il codice dell'istruzione è espressa in linguaggio mnemonico (3 caratteri) : COP consente i 56 codici operativi del 6502; la parte OPE ha un formato variabile che segue il modo di indirizzamento. */300: LDA#00 RETURN* restituisce *300-A9 00 LDA#00* questa istruzione consente di porre a zero il registro accumulatore.
spazio COP OPE	Assembla l'istruzione all'indirizzo seguente.
FORMATO della parte relativa all'operando OPE	Se l'indirizzo è di due cifre agisce in pagina zero.
#$ valore	Indirizzamento immediato.
$ indirizzo	Indirizzamento assoluto (pagina zero oppure che non segue COP).
$ indirizzo, X	Indirizzo indicizzato da X (pagina zero oppure che non segue COP).
$ indirizzo, Y	Indirizzo indicizzato da Y (pagina zero oppure che non segue COP).
$ indirizzo	Indirizzamento relativo (per le istruzioni di salto all'indirizzo).
($ indirizzo, X)	Indirizzamento indiretto pre-indicizzato da X.
($ indirizzo, Y)	Indirizzamento indiretto pre-indicizzato da Y.

Comando	*Definizione - esempio*
($ indirizzo)	Indirizzamento indiretto (per l'istruzione di salto JMP). */300:LDA $00,X* *300-B5 00* */ spazio DEX* *303-CA* */ spazio BNE $300* *304-D0 FB*
$	Chiamata al Monitor */$300.305* per verificare il contenuto di una zona assemblata.

Salvataggio - caricamento dei programmi

LOAD "NOME",D1	Carica in memoria RAM il programma "NOME" dal dischetto posto nell'unità 1.
SAVE "NOME",D2	Salva un programma BASIC che risiede in memoria RAM sul dischetto posto nell'unità 2 con il nome "NOME".
BLOAD "BINARIO"	Carica in memoria RAM il file binario "BINARIO" dal dischetto all'indirizzo assoluto registrato in testa al file. Questo indirizzo e la lunghezza del file sono nella memoria RAM all'indirizzo $AA72 e $AA60 prima del caricamento.
BSAVE "BINARIO",A$300,L$7F	Salva il programma in linguaggio macchina posto in memoria partendo dall'indirizzo assoluto $300 per una lunghezza di $7F byte sul dischetto corrente sotto il nome di "BINARIO".
BSAVE "IMM",A$2000,L$1FF8	Salva l'immagine grafica in alta risoluzione contenuta nella pagina 1 sul dischetto con il nome "IMM".
HGR:BLOAD "IMM"	Restituisce sullo schermo grafico ad alta risoluzione i punti registrati nel file "IMM".
RUN "NOME"	Carica in memoria RAM da disco ed esegue il programma che si chiama "NOME".
BRUN "BINARIO"	Carica in memoria RAM da disco ed esegue il programma scritto in linguaggio macchina che si chiama "BINARIO".
CHAIN "NOME"	Carica in memoria RAM da disco un programma scritto in BASIC Integer senza cancellare la zona delle variabili del programma precedente. Il programma "NOME" non deve avere variabili comuni ridimensionate.

D$=CHR$(13)+CHR$(4)

File sequenziali (battere T)

PRINT D$"OPEN FI"	Apre sul disco selezionato un file denominato FI. Posiziona il puntatore all'inizio del file sequenziale.
PRINT D$"OPEN"F$",D1"	Apre sul disco posto nell'unità 1 un file di nome variabile F$.
PRINT D$"READ FI"	Predispone una operazione di lettura all'inizio del file sequenziale FI aperto con il comando OPEN.
INPUT A$	Legge nel file FI una stringa di caratteri che sarà memorizzata in memoria RAM sotto la variabile A$. Il puntatore è posto all'inizio del campo dei dati seguente.
GET C$	Legge un solo carattere e sposta il puntatore di un carattere.
PRINT D$"POSITION FI,R"P	Posiziona il puntatore dopo il P-iesimo RETURN posto dopo la posizione attuale.
PRINT D$"WRITE FI"	Prepara un'operazione di scrittura sul file FI dove si trova il puntatore.
PRINT X$	Scrive nel file FI la stringa X$.
PRINT Y$	Scrive la stringa Y$ separata dalla stringa X$, precedentemente registrata, da un carattere RETURN.
PRINT CHR$(4)"CLOSE FI"	Chiude il file FI salvando su disco i parametri relativi alla operazione di OPEN effettuata sul file contenente l'ultimo settore utilizzato.
PRINT D$"APPEND FI"	Ritrova il file FI e posiziona il puntatore alla fine del file. Permette di scrivere dati e aggiunte alla fine del file.
PRINT D$"READ FI,B17"	Posiziona il puntatore al byte 17 per una successiva lettura (0 è il primo byte).

COMANDI RELATIVI AL SISTEMA OPERATIVO DOS

D$=CHR$(13)+CHR$(4)

File ad accesso diretto (battere T)

PRINT D$"OPEN PIP,L21"

Apre il file PIP su disco prevedendo record di lunghezza costante; in questo caso i 21 byte previsti comprendono i RETURN assunti come fine del record.

APERTURA

PRINT D$"OPEN"F$",L"N",D2"

Apre un file di nome variabile F$ di lunghezza uguale alla variabile N sul disco posto nell'unità 2.

PRINT D$"READ PIP,R"I

Posiziona il puntatore all'inizio dell'I-iesimo record per una successiva lettura.

INPUT A$

Recupera in A$ il contenuto del'I--iesimo record nel file PIP.

GET C$

LETTURA

Legge il carattere posto sotto il puntatore nel record I.

PRINT D$"WRITE PIP,R"J

Posiziona il puntatore all'inizio del J-iesimo record per una successiva scrittura.

PRINT X$

SCRITTURA

Scrive la stringa X$ nel record J.

PRINT Y$

Scrive la stringa Y$, separata dalla stringa precedentemente registrata, con RETURN.

CHIUSURA

PRINT CHR$(4)"CLOSE PIP"

Chiude il file PIP salvando i parametri relativi.

PRINT D$"READ PIP,R0,B"K

Posiziona il puntatore al byte K del record 0 per una ulteriore lettura.

PRINT CHR$(4)

Annulla l'effetto di un comando DOS, come READ, per esempio per dare all'istruzione INPUT il senso di una impostazione da tastiera.

UTIL. PER CARICARE PARTE DI PROGRAMMA

Comandi diversi

CATALOG D2

Visualizza l'elenco dei programmi e dei file registrati sul dischetto posto nell'unità 2.

* file protetti dalle cancellazioni accidentali.
I, A, T, B tipi di file:
  I : BASIC Integer
  A : BASIC Applesoft
  T : File TEXT sequenziali o diretti
  B : File binari, dati o programmi in linguaggio macchina.
002 : Numero dei settori occupati dal file (modulo 256).

Un settore comprende 256 byte utili.
Una traccia comprende 13 o 16 settori secondo il DOS impiegato (3.2 o 3.3).
Un dischetto comprende 35 tracce delle quali 31 sono utilizzabili dall'utente e 4 riservate al sistema.

Il numero massimo di nomi di file che possono essere registrati su disco è 84 in DOS 3.2 e 105 in DOS 3.3.

MON C,I,O

Visualizza i comandi, le entrate e le uscite come sono ricevute o inviate dal sistema operativo durante l'esecuzione di un programma.
Il comando si annulla con RESET.

NOMON C,I,O

Annulla il comando MON.

PRINT D$"PR#"S

Attiva la periferica collegata allo slot "S" per l'esecuzione della successiva istruzione PRINT.
Se l'interfaccia relativa alla stampante risiede nello slot 1:
PRINT D$"PR#1".

PRINT D$"PR#0"

Disattiva la periferica di uscita lasciando in linea solo lo schermo.

PRINT D$"IN#"S

Connette la periferica collegata allo slot "S" per l'acquisizione di dati da inviare al calcolatore.

COMANDI RELATIVI AL SISTEMA OPERATIVO DOS

PRINT D$"IN#0"

Limita alla tastiera la possibilità di input verso il calcolatore.

MAXFILE 4

Prevede l'utilizzo di 4 buffer di input/output aperti contemporaneamente in parallelo corrispondenti a 4 files aperti disponibili in memoria.
Ciascun buffer occupa 595 byte.
Per default il sistema riserva 3 zone riservate ai buffer.
Questo comando dev'essere eseguito prima del caricamento e dell'esecuzione del programma.

VERIFY NOME

Verifica la buona registrazione fisica del programma o del file NOME.
Se un settore del dischetto utilizzato è danneggiato sarà visualizzato il messaggio I/O ERROR (la verifica avviene anche automaticamente dopo il comando SAVE).

Elaborazione dei file

INIT HELLO [,V254]

Procedura di inizializzazione di un dischetto vergine. Il programma BASIC che si trova in memoria RAM è caricato sul dischetto con il nome HELLO insieme al sistema operativo.
Il dischetto porta un numero di volume che può essere utile per eventuali controlli.
Il formato con il quale viene inizializzato il dischetto dipende dal sistema operativo che risiede in memoria RAM.

DOS 3.2 : 13 settori per traccia
DOS 3.3 : 16 settori per traccia
Lo spazio disponibile è:
DOS 3.2 : 103.168 byte per dischetto
DOS 3.3 : 126.976 byte per dischetto
Il sistema operativo registrato sul dischetto è del tipo "schiavo" e contiene solo il programma di rilocazione per adattarsi nel modo migliore alla configurazione del sistema.

DELETE NOME	Sopprime il file NOME dal dischetto se il file non è LOCK (protetto contro la cancellazione).
RENAME VECCHIO,NUOVO	Cambia il nome del file o di un programma.
LOCK NOME	Protegge il file NOME contro le cancellature accidentali che possono essere provocate dai comandi DELETE, SAVE oppure WRITE.
UNLOCK NOME	Toglie la protezione contro le cancellature dal file NOME.

File di comando

LIST	
PRINT D$"OPEN CMD"	Creazione di un file
PRINT D$"WRITE CMD"	
PRINT "PR#1"	Registrazione degli ordini
PRINT CHR$(9)"60 N"	
PRINT "LIST"	
PRINT "PR#0"	
PRINT D$"CLOSE CMD"	Chiusura del file
RUN	Realizzazione del file
EXEC CMD	Esecuzione automatica dei comandi registrati nel file CMD nel nostro esempio stampa di un listing sulla stampante con un formato di 60 caratteri per linea.

Tutte le istruzioni e i comandi eseguibili in modo diretto possono appartenere ad un file del tipo appena esaminato.

CONVERSIONE ESADECIMALE/DECIMALE/ESADECIMALE

Dei primi 256 numeri da $00 a $FF

$00	:0	1	2	3	4	5	6	7
$08	:8	9	10	11	12	13	14	15
$10	:16	17	18	19	20	21	22	23
$18	:24	25	26	27	28	29	30	31
$20	:32	33	34	35	36	37	38	39
$28	:40	41	42	43	44	45	46	47
$30	:48	49	50	51	52	53	54	55
$38	:56	57	58	59	60	61	62	63
$40	:64	65	66	67	68	69	70	71
$48	:72	73	74	75	76	77	78	79
$50	:80	81	82	83	84	85	86	87
$58	:88	89	90	91	92	93	94	95
$60	:96	97	98	99	100	101	102	103
$68	:104	105	106	107	108	109	110	111
$70	:112	113	114	115	116	117	118	119
$78	:120	121	122	123	124	125	126	127
$80	:128	129	130	131	132	133	134	135
$88	:136	137	138	139	140	141	142	143
$90	:144	145	146	147	148	149	150	151
$98	:152	153	154	155	156	157	158	159
$A0	:160	161	162	163	164	165	166	167
$A8	:168	169	170	171	172	173	174	175
$B0	:176	177	178	179	180	181	182	183
$B8	:184	185	186	187	188	189	190	191
$C0	:192	193	194	195	196	197	198	199
$C8	:200	201	202	203	204	205	206	207
$D0	:208	209	210	211	212	213	214	215
$D8	:216	217	218	219	220	221	222	223
$E0	:224	225	226	227	228	229	230	231
$E8	:232	233	234	235	236	237	238	239
$F0	:240	241	242	243	244	245	246	247
$F8	:248	249	250	251	252	253	254	255
	8	9	A	B	C	D	E	F

32256

CODICI DI TASTIERA

SHIFT	A1	A2	A3	A4	A5	A6	A7	A8	A9	B0	AA	BD	CTRL
	A1	A2	A3	A4	A5	A6	A7	A8	A9	B0	AA	BD	
	! 1	" 2	# 3	$ 4	% 5	& 6	7	(8	) 9	0	* :	= -	RESET
	B1	B2	B3	B4	B5	B6	B7	B8	B9	B0	BA	AD	
	B1	B2	B3	B4	B5	B6	B7	B8	B9	B0	BA	AD	CTRL

SHIFT	9B	91	97	85	92	94	99	95	89	8F	80		8D	CTRL
	9B	D1	D7	C5	D2	D4	D9	D5	C9	CF	C0		8D	
	ESC	Q	W	E	R	T	Y	U	I	O	@ P	REPT	RETURN	
	9B	D1	D7	C5	D2	D4	D9	D5	C9	CF	D0		8D	
	9B	91	97	85	92	94	99	95	89	8F	90		8D	CTRL

SHIFT		81	93	84	86	87	88	8A	8B	8C	AB	88	95	CTRL
		C1	D3	C4	C6	C7	C8	CA	CB	CC	AB	88	95	
	CTRL	A	S	D	F	BELL G	H	J	K	L	+ ;	<-	->	
		C1	D3	C4	C6	C7	C8	CA	CB	CC	BB	88	95	
		81	93	84	86	87	88	8A	8B	8C	BB	88	95	CTRL

SHIFT		9A	98	83	96	82	9E	9D	BC	BE	BF	CTRL
		DA	D8	C3	D6	C2	DE	DD	BC	BE	BF	
	SHIFT	Z	X	C	V	B	N	M	< ,	> .	? /	SHIFT
		DA	D8	C3	D6	C2	CE	CD	AC	AE	AF	
		9A	98	83	96	82	8E	8D	AC	AE	AF	CTRL

SHIFT	A0	CTRL
	A0	
	SPAZIO	
	A0	
	A0	CTRL

I codici ASCII estesi (7 bit più uno) dei tasti in **tastiera** in **esadecimale.**

Normale

```
LIST

10  HOME
20  FOR I = 160 TO 191
30  PRINT  CHR$ (I);
40  NEXT I
50  PRINT
60  FOR I = 192 TO 223
70  PRINT  CHR$ (I);
80  NEXT I
```

```
RUN
 !"#$%&'()*+,-./0123456789:;<=>?
@ABCDEFGHIJKLMNOPQRSTUVWXYZ°\é^_
```

```
CALL-151

*400.41F

0400- A0 A1 A2 A3 A4 A5 A6 A7
0408- A8 A9 AA AB AC AD AE AF     1ª linea di schermo
0410- B0 B1 B2 B3 B4 B5 B6 B7
0418- B8 B9 BA BB BC BD BE BF

*480.49F

0480- C0 C1 C2 C3 C4 C5 C6 C7
0488- C8 C9 CA CB CC CD CE CF     2ª linea di schermo
0490- D0 D1 D2 D3 D4 D5 D6 D7
0498- D8 D9 DA DB DC DD DE DF
```

CODICI DI SCHERMO

Inverso

```
 RUN                                          1ª linea
 /"#$%&'(°\_,-./0123456789:;<=>?               2ª linea
@ABCDEFGHIJKLMNOPQRSTUVWXYZ°\è^_
```

```
 CALL-151
```

```
*400.41F

0400- 20 21 22 23 24 25 26 27
0408- 28 29 2A 2B 2C 2D 2E 2F
0410- 30 31 32 33 34 35 36 37        1ª linea
0418- 38 39 3A 3B 3C 3D 3E 3F
```

```
*480.49F

0480- 00 01 02 03 04 05 06 07
0488- 08 09 0A 0B 0C 0D 0E 0F
0490- 10 11 12 13 14 15 16 17        2ª linea
0498- 18 19 1A 1B 1C 1D 1E 1F
```

Flash

```
 RUN
@abcdefghijklmnopqrstuvwxyzàòè
@ABCDEFGHIJKLMNOPQRSTUVWXYZ°\é^_


 CALL-151

*400.41F

0400- 60 61 62 63 64 65 66 67
0408- 68 69 6A 6B 6C 6D 6E 6F          1ª linea
0410- 70 71 72 73 74 75 76 77
0418- 78 79 7A 7B 7C 7D 7E 7F


*480.49F

0480- 40 41 42 43 44 45 46 47
0488- 48 49 4A 4B 4C 4D 4E 4F
0490- 50 51 52 53 54 55 56 57          2ª linea
0498- 58 59 5A 5B 5C 5D 5E 5F
```

Codici di schermo

In uscita sulla stampante, i codici da $60 a $7F corrispondono ai codici ASCII relativi ai caratteri minuscoli, questa è la ragione per la quale sono stati stampati in minuscolo dalla stampante quando sono stati attivati come caratteri lampeggianti.

Il gruppo ! " # ... 0 1 ... 9 ... ? è quello delle **cifre**.
Il gruppo @ A B C ..... Z .. è quello delle **lettere**.

LETTERE IN MODO INVERSO	$00 - $1F
CIFRE IN MODO INVERSO	$20 - $3F
LETTERE IN MODO LAMPEGGIANTE	$40 - $5F
CIFRE IN MODO LAMPEGGIANTE	$60 - $7F
CIFRE IN MODO NORMALE	$A0 - $BF
LETTERE IN MODO NORMALE	$C0 - $DF

ASCII (POSITIVO) (BIT 7=0)

Codici dei caratteri in esadecimale

00	:		ᶜ	ᶜA	ᶜB	ᶜC	ᶜD	ᶜE	ᶜF	ᶜG
08	:	ᶜH	ᶜI	ᶜJ	ᶜK	ᶜL	ᶜM	ᶜN	ᶜO
10	:	ᶜP	ᶜQ	ᶜR	ᶜS	ᶜT	ᶜU	ᶜV	ᶜW
18	:	ᶜX	ᶜY	ᶜZ	esc	.	ᶜSM	ᶜ^	.
20	:	esp	!	"	#	$	%	&	'
28	:	(	)	*	+	,	-	.	/
30	:	0	1	2	3	4	5	6	7
38	:	8	9	:	;	<	=	>	?
40	:		A	B	C	D	E	F	G
48	:	H	I	J	K	L	M	N	O
50	:	P	Q	R	S	T	U	V	W
58	:	X	Y	Z	[	\	]	^	_
60	:	`	a	b	c	d	e	f	g
		.	.	.	.	.	.	.	.
68	:	h	i	j	k	l	m	n	o
		.	.	.	.	.	.	.	.
70	:	p	q	r	s	t	u	v	w
		.	.	.	.	.	.	.	.
78	:	x	y	z	{	|	}		
		.	.	.	.	.	.		

A per <u>CTRL-A</u>

ᶜH = backspace, ←
ᶜJ = salto di linea (LF Line Feed)
ᶜL = salto pagina (FF Form Feed)

ᶜM = ritorno carello, <u>RETURN</u>
ᶜU = NAK, →
ESC = escape
ᶜSM= CTRL SHIFT M

ᶜG = BELL (campanello)

] = SHIFT M

. = non possono essere introdotti da tastiera.

Codici dei caratteri in esadecimale

A0 :		!	"	#	$	%	&	'
A8 :	(	)	*	+	,	-	.	/
B0 :	0	1	2	3	4	5	6	7
B8 :	8	9	:	;	<	=	>	?
C0 :	@	A	B	C	D	E	F	G
C8 :	H	I	J	K	L	M	N	O
D0 :	P	Q	R	S	T	U	V	W
D8 :	X	Y	Z	[	\	]	^	_
E0 :	`	a	b	c	d	e	f	g
E8 :	h	i	j	k	l	m	n	o
F0 :	p	q	r	s	t	u	v	w
F8 :	x	y	z	{	|	}		

Caratteri di controllo

80 :	^@	^A	^B	^C	^D	^E	^F	^G
88 :	^H	^I	^J	^K	^L	^M	^N	^O
90 :	^P	^Q	^R	^S	^T	^U	^V	^W
98 :	^X	^Y	^Z	esc	.	^SM	^^	.

^U = →, '95'
^H = ←, '88'
^M = <u>RETURN</u>, '8D'

CONVERSIONE ESADECIMALE/DECIMALE

Numeri di 4 cifre esadecimali $H3 H2 H1 H0

COEFF	H3	H2	H1	H0
0	0	0	0	0
1	4096	256	16	1
2	8192	512	32	2
3	12288	768	48	3
4	16384	1024	64	4
5	20480	1280	80	5
6	24576	1536	96	6
7	28672	1792	112	7
8	32768	2048	128	8
9	36864	2304	144	9
A	40960	2560	160	10
B	45056	2816	176	11
C	49152	3072	192	12
D	53248	3328	208	13
E	57344	3584	224	14
F	61440	3840	240	15

CONVERSIONE ESADECIMALE/DECIMALE

Il numero decimale è ottenuto facendo la somma dei valori presi all'intersezione della linea relativa alle cifre esadecimali e la colonna alla posizione delle cifre nel numero esadecimale.

```
Esempio:    $AFF6 diventa   40960 (A in H3)
                          +  3840 (F in H2)
                          +   240 (F in H1)
                          +     6 (6 in H0)
                          -------
                            45046
```

MESSAGGI DI ERRORE

APPLESOFT

I messaggi di errore, in un programma Applesoft, hanno la seguente forma:

? messaggio ERROR IN numero di linea

"messaggio" è il nome dell'errore.
Il numero di linea è quello relativo alla linea che contiene l'istruzione nella quale è stato riscontrato l'errore. Gli errori sono rilevati solo quando si procede all'esecuzione del programma.

Quando è riscontrato un errore, l'interprete BASIC Applesoft provoca **l'arresto** del programma e **la visualizzazione** del messaggio relativo. Le variabili e le istruzioni non vengono modificate, ma il programma non può procedere. I contatori dei cicli FOR-NEXT sono rimessi a zero e l'eventuale istruzione GOSUB è annullata.

Grazie all'istruzione ONERR GOTO e ad un sottoprogramma di trattamento degli errori un programma può malgrado tutto proseguire normalmente.

Le istruzioni date in modo diretto (senza numero di linea) possono provocare l'emissione di un messaggio di errore, in tal caso il messaggio non conterrà l'indicazione del numero di linea.

Ciascun tipo di errore è associato ad un codice che è posto,al momento dell'errore, all'indirizzo decimale 222 (oppure, in esadecimale, $DE).

Il numero di linea dove l'errore si è verificato figura negli indirizzi decimali 218 e 219 (oppure, in esadecimale, $DA e $DB). Il valore del puntatore TXPTR nell'istruzione errata è posto agli indirizzi decimali 220 e 221 (oppure, in esadecimale, $DC e $DD). Il valore dei puntatori relativi allo stack, al momento dell'errore, è conservato all'indirizo decimale 223 (oppure, in esadecimale, $DF). Queste informazioni devono essere ripristinate prima di procedere al trattamento dell'errore; questo può essere fatto grazie al seguente programma:

```
300   68      PLA
301   A8      TAY
302   68      PLA
303   A6 DF   LDX $DF
305   9A      TXS
306   48      PHA
307   98      TYA
308   48      PHA
309   60      RTS
```

Tale sottoprogramma può essere scritto, nella memoria del calcolatore, per mezzo delle seguenti istruzioni:

```
0 POKE 216,0 : POKE 768,104 : POKE 769,168 : POKE 770,104 :
POKE 771,166 : POKE 772,223 : POKE 773,154 : POKE 774,72 :
POKE 775,152 : POKE 776,72 : POKE 777,96
```

All'indirizzo decimale 216 figura l'indicazione di attivazione ($80) oppure la disattivazione ($00) dell'istruzione ONERR GOTO.

Segue un programma che tien conto di eventuali errori e del conseguente arresto del programma:

```
   0  memorizzazione del programma che segue
   1  ONERR GOTO 1000 : attivazione del sistema
  10  normale svolgimento del programma
 999  END
1000  CALL 768 : esecuzione del sottoprogramma
1010  IF PEEK(222)=5 THEN prende in considerazione l'errore 5
1020  verifica di un altro tipo di errore
1030  RESUME per tornare all'istruzione errata se necessario
```

Il sottoprogramma che controlla lo svolgimento del programma in caso di errore è posto all'indirizzo esadecimale $D412. Provoca l'esecuzione del sottoprogramma HANDLERR all'indirizzo esadecimale $F2E9 se ONERR è attivo. HANDLERR sistema le locazioni di memoria $DA e $DF.

Se ONERR non è stato utilizzato, il sottoprogramma $D412 arresta l'esecuzione del programma e visualizza il messaggio di errore.

Nell'interprete Applesoft, la tabella dei messaggi di errore è posizionata partendo dall'indirizzo $D260 e i loro codici corrispondono alla loro posizione in questa tabella.

```
NEXT WITHOUT FORSYNTAXRETURN WITHOUT GO
SUBOUT OF DATAILLEGAL QUANTITYOVERFLOWOU
T OF MEMORYUNDEF'D STATEMENTBAD SUBSCRIP
TREDIM'D ARRAYDIVISION BY ZEROILLEGAL DI
TOO COMPLEXCAN'T CONTINUEUNDEF'D FUNCTIO
N ERROR IN
BREAK
```

Elenco dei messaggi di errore in ordine alfabetico

Codice	*Messaggio*	*Origine*	*Commento*
107	?BAD SUBSCRIPT (indice errato)	DIM	Tentativo di chiamare un elemento di una tabella di indice superiore al limite fissato con l'istruzione DIM. Esempio: DIM A(15) con A(20) oppure ancora con un numero d'indice differente da quelli previsto da DIM. L'Applesoft dimensiona automaticamente a 11 le variabili indicizzate non dichiarate.
210	CAN'T CONTINUE (non posso continuare)	CONT	Impossibilità di riprendere l'esecuzione del programma con CONT. In caso di errore oppure di inserimento o modifica di una istruzione. In certi casi si può fare ripartire il programma con un GOTO ad un numero di linea.
133	?DIVISION BY ZERO (divisione per zero)	/0	Può derivare da una variabile non inizializzata con un valore diverso da zero.
	?EXTRA IGNORED (troppi dati) ignorati	INPUT	Se i dati, separati da virgole, sono in numero superiore a quello delle variabili previste. Il programma prosegue comunque.
191	?FORMULA TOO COMPLEX (formula troppo complessa)	IF "stringa di caratteri" THEN	Il test non può essere richesto per più di due volte nello stesso programma.

Codice	*Messaggio*	*Origine*	*Commento*
149	?ILLEGAL DIRECT (illegale in modo diretto)	Modo diretto	Le istruzioni INPUT, GET DEF FN e DATA non possono essere usati in modo diretto.
53	?ILLEGAL QUANTITY (valore errato)	Funzione matematica	I parametri forniti in una funzione superano i limiti permessi. - l'indice di una variabile è negativo - l'argomento di LOG è negativo o nullo - l'argomento di SQR è negativo. A potenza di B se A è negativo e B non è intero.
53	?ILLEGAL QUANTITY	MID$ LEFT$ RIGHT$	La lunghezza o l'indice di posizionamento non è compreso tra 1 e 255.
		CHR$	Il codice non è compreso tra 0 e 255.
		ASC	Il carattere è di lunghezza 0.
		CALL	L'indirizzo non è compreso tra -65535 e +65535.
		POKE	L'indirizzo non è compreso tra -65535 e +65535; il valore non è compreso tra 0 e 255.
		HIMEM:	L'indirizzo non è compreso tra -65535 e +65535.
		HPLOT DRAW PLOT,VLIN HLIN PDL	X,Y<0 o X>278 e Y>191 X,Y<0 o X>278 e Y>191 X,Y<0 o X,Y>39 X<0 o X>255

Codice	Messaggio	Origine	Commento
		HTAB VTAB SPC TAB(	X<0 o X>255 X<0 o X>24 X<0 o X>255 X<0 o X>255
		ON...GOTO ON...GOSUB	L'indice non deve superare 255 e non deve essere inferiore a 0. Se il valore dell'indice è nullo o più elevato dei numeri del numero di linea specificato, esecuzione procede alla linea successiva.
0	NEXT WITHOUT FOR (NEXT senza FOR)	FOR, NEXT	Cicli FOR..NEXT sono mal posizionati. Esempio: FOR X=1 TO ... FOR Y=1 TO ... PRINT X,Y NEXT : NEXT Y (scrivere NEXT Y:NEXT Y) Manca un FOR per un NEXT isolato.
42	?OUT OF DATA (dati mancanti)	READ	Prova di esecuzione di una istruzione READ quando tutti i DATA sono già stati letti. Prevedere un carattere per testare la fine dei dati o una variabile di conteggio oppure eseguire RESTORE per rileggere i dati all'inizio dei DATA.
		RECALL STORE	Non utilizzare variabili il cui nome comincia con RECALL o STORE.
77	?OUT OF MEMORY (memoria mancante)	DIM GOSUB HIMEM:	Non superare il numero massimo di indice: 88. Non può gestire oltre 24 livelli di chiamata. Non fissare valori troppo bassi.

Codice	*Messaggio*	*Origine*	*Commento*
		LOMEM:	Non fissare valori troppo alti oppure sotto il valore attuale. Il programma è troppo grande o le variabili troppo numerose.
66	?OVERFLOW ERROR (supero di capacità'		Risultato superiore a 1.7E38.
		Numero reale	Un numero reale è memorizzato con un byte di esponente e 4 byte per la mantissa. Se il risultato è inferiore a 2.9E-39, equivale a 0 senza messaggio di errore.
		STR$	Se il numero da convertire in stringa di caratteri è troppo grande.
		VAL	Se il valore assoluto del numero cercato è superiore a 1E38 o se il numero contiene più di 13 cifre.
120	?REDIM'D ARRAY (matrice ridimensionata)	DIM	Una stessa matrice non può essere ridimensionata due volte.
	?REENTER (rifare l'impostazione dei dati)	INPUT	Si è eseguito un INPUT alfanumerico; si devono ridare tutti i valori numerici attesi dall'istruzione INPUT.
22	?RETURN WITHOUT (RETURN senza GOSUB)	RETURN	Un sottoprogramma è stato posto prima della fine logica del programma oppure è stata dimenticata l'istruzione END. Nel trattamento di un errore, riprendere da un sottoprogramma senza eseguire GOSUB.

Codice	Messaggio	Origine	Commento
176	?STRING TOO LONG (stringa di caratteri troppo lunga)	X$ LEN VAL PRINT	Non creare una stringa di caratteri per la concatenazione la cui lunghezza superi 255. Se l'argomento è una stringa di lunghezza totale superiore a 255. A$+B$ ha più di 255 caratteri (scrivere PRINT AB).
16	?SYNTAX (errore di sintassi)		Istruzione incomprensibile per l'interprete Applesoft. - parentesi non chiuse - caratteri illegali - punteggiatura errata - errore di ortografia in una parola chiave
		ASC	Su "CRTL @" o CHR$(0).
		CONT	Se i dati relativi a INPUT sono interrotti con CTRL-C e si tenta di ripartire con CONT.
16	?SYNTAX (errore di sintassi)	DATA	Una stringa di caratteri contenente ? non è accettata.
16	?SYNTAX (errore di sintassi)	DEL	Dev'essere seguito da 2 numeri di linea di ordine crescente.
		FOR...NEXT	Non utilizzare una variabile di tipo intero (%) come indice di ciclo.
		HGR HGRZ TEXT	Non utilizzare queste parole chiavi come i primi caratteri di una variabile (sarebbero eseguiti prima del messaggio di errore).

Codice	*Messaggio*	*Origine*	*Commento*
		IF...THEN	Manca il THEN dell'IF corrispondente.
		LIST, Q	Visualizza il programma completo seguito da ?SYNTAX.
		RECALL STORE SHLOAD	Non utilizzare queste parole chiavi come i primi caratteri di una variabile.
		RESUME	L'istruzione viene riscontrata prima che sia prodotto un errore. Può essere un errore fatale.
		N°d'istruzione errato.	Se il carattere O è dato al posto del numero 0 o se la lettera I è data al postop del numero 1.
163	?TYPE MISMATCH (disaccordo tra numerico e alfanumerico)	LET MID$ LEFT$ RIGHT$	- una variabile stringa non può ricevere un numerico e viceversa - errato il tipo di argomento.
224	?UNDEF'D FUNCTION (funzione non defita)		Riferito ad una funzione per la quale non esiste l'istruzione DEF FN.
90	?UNDEF'D STATEMENT (statement indefinito)	GOTO GOSUB ON..GOTO RUN THEN	Rinvio ad un numero di linea inesistente.

MESSAGGI DI ERRORE RELATIVI ALLA GESTIONE DEI DISCHI

Nel sistema operativo dei dischi caricato in memoria RAM (in un sistema da 48 Kbyte), la tabella dei messaggi di errore è posta a partire dall'indirizzo $A971. Il primo messaggio è 'RETURN', 'BELL , 'RETURN'. I seguenti sono:

```
LANGUAGE NOT AVAILABLERANGE ERRORWRI
TE PROTECTEDEND OF DATAFILE NOT FOUNDVOL
UME MISMATCHI/O ERRORDISK FULLFILE LOCKE
DSYNTAX ERRORNO BUFFERS AVAILABLEFILE TY
PE MISMATCHPROGRAM TOO LARGENOT DIRECT C
OMMAND
```

Nella zona seguente, da $AA3F a $AA4F, sono registrate le posizioni di inizio di ciascun messaggio contenuto nella tabella precedente:

0	3	25	25	36	51	62	76	91	100	109	120	132	152	170	187	DEC.
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	CODICE
				$24	$33											ESAD.

Esempio: il messaggio WRITE PROTECTED è il quarto della lista, il suo codice di errore è 4 e il testo è memorizzato dall'indirizzo $A971+$24 all'indirizzo $A971+$32.

Una corretta analisi consentirà la traduzione in italiano dei messaggi di errore inviati dal sistema operativo.

MESSAGGI DI ERRORE DEL SISTEMA OPERATIVO
(in ordine alfabetico)

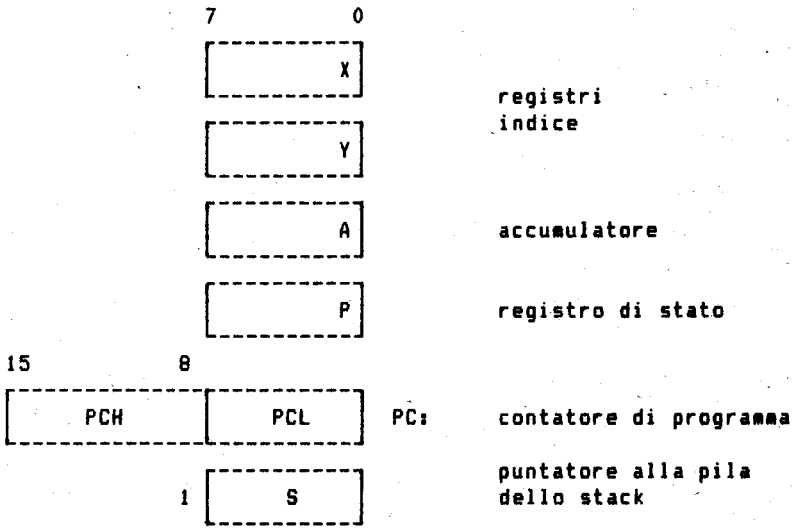
Codice	*Messaggio*	*Origine*	*Commento*
9	DISK FULL (disco pieno)	SAVE WRITE	La directory è stata riempita oppure tutti i settori sono stati utilizzati.
5	END OF DATA (fine dei dati)	INPUT	I dati sono insufficienti per soddisfare l'istruzione INPUT.
		APPEND READ	Dopo questa istruzione è ammessa solo l'istruzione WRITE.
		POSITION READ	La POSITION specificata non corrisponde ad alcun dato registrato.
		EXEC F,Rr	Se r corrisponde al 21mo campo dopo la fine del file.
		READ F,Rr	Se r corrisponde ad una registrazione non ancora effettuata (codice 0).
10	FILE LOCKED (file o programma protetto)	SAVE DELETE BSAVE WRITE	Al file è associato un asterisco nel CATALOG e non può essere modificato in scrittura (salvo con APPEND).
6	FILE NOT FOUND (file non trovato)	LOAD RUN BLOAD BRUN DELETE	Il file chiesto non esiste sul dischetto. Verificare l'ortografia del suo nome.
13	FILE TYPE MISMATCH (disaccordo sul tipo di file)	LOAD BLOAD RUN BRUN	Un file di tipo T o B non può essere caricato con LOAD o RUN; un file di tipo I o A non può essere chiamato con BLOAD o BRUN.
13	FILE TYPE MISMATCH (disaccordo sul tipo di file)	OPEN READ POSITION WRITE APPEND EXEC CLOSE	Questi comandi sono operativi solo su file di dati (tipo T).

Codice	*Messaggio*	*Origine*	*Commento*
		CHAIN	Un programma in BASIC Applesoft non può essere 'CHAIN' con questo solo comando che riguarda solo i programmi BASIC Integer.
8	I/O ERROR (errore di Input/ Output)	Tutti i comandi	- lettura senza disco - n. di slot (senza il controller dei dischi) - disco rovinato - disco non inizalizzato - porta di lettura aperta
		VERIFY	Se c'è un errore dopo la verifica che un file è registrato correttamente o meno.
1	LANGUAGE NOT AVAILABLE (interprete BASIC non presente)	LOAD FP INT	Un programma BASIC non può essere eseguito se non è presente il relativo interprete.
		APPLE][PLUS	Non ha l'interprete BASIC Integer standard.
		SCHEDA LINGUAGGI	Se la scheda madre contiene in ROM un interprete, la scheda linguaggi potrà caricare l'altro.
12	NO BUFFER AVAILABLE (troppi file aperti in memoria RAM)	MAXFILES n	Il numero massimo è 16 (il sistema ne utilizza 1 per ciascun comando). Per default 3 file.
15	NOT DIRECT COMMAND (comando diretto illegale)	OPEN READ WRITE APPEND POSITION	Non possono essere utilizzati in modo diretto. - scrivere un programma che contiene questi comandi in una PRINT.

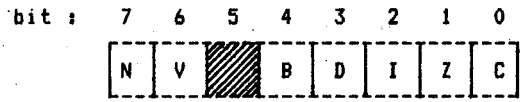
MESSAGGI DI ERRORE DEL SISTEMA OPERATIVO

Codice	*Messaggio*	*Origine*	*Commento*
14	PROGRAM TOO LARGE (programma troppo grande)	LOAD RUN	HIMEM è troppo bassa (il sistema operativo compara il numero dei settori del programma con il byte più significativo di HIMEM).
2, 3	RANGE ERROR (valore errato)		Min - Max
		V	Volume 0 - 245
		D	Drive 0 - 2
		S	Slot 1 - 7
		L	Lunghezza 1 - 32767
		R	Numero 0 - 32767
		B	Numero 0 - 32767
		A	Indirizzo 0 - 65535
		MAXFILES	Numero dei file aperti 1 - 16
11	SYNTAX ERROR (errore di sintassi in un comando di DOS)	INT EXEC	Comando senza parametri istruzione BASIC non valido.
		Àa	a è negativo
		L1	1 > 65535
		IN#s PR#s	s non può essere superiore a 7.
7	VOLUME MISMATCH (disaccordo con il numero di volume)	V	Il volume del disco corrente è diverso da quello del disco richiesto. Se la richiesta è fatta con V0, la verifica del volume non potrà essere effettuata.
4	WRITE PROTECTED (protetto in scrittura)		La tacca sul disco è coperta, il file è accessibile solo in lettura. Il disco può essere inserito al contrario. Il dischetto SYSTEM MASTER è sempre protetto.

REGISTRI INTERNI DEL 6502



Dettaglio sul registro di stato P



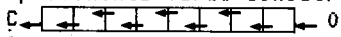
N	segno	D	modo decimale
V	overflow	I	interrupt
5	inutilizzato	Z	zero
B	comando di Break	C	carry

SET DI ISTRUZIONE DEL 6502

ADC : Addizione con riporto (ADd whit Carry) : $A \leftarrow A + M + C$
Si aggiunge all'accumulatore la memoria specificata più il bit di riporto.
Opera in modo binario o decimale. Agisce su N, V, Z, C.

AND : And logico (AND) : $A \leftarrow A \wedge M$
Esegue l'and bit per bit nell'accumulatore e nella memoria.
Agisce su $\overline{N}$, Z.

ASL : Spostamento verso sinistra (Arithmetic Shift Left)



Sposta a sinistra il contenuto dell'accumulatore o di una memoria. Agisce su N, Z, C.

BCC : Esegue il salto se il carry è uguale a 0 (Branch on Carry Clear). Se il bit C = 0, si salta all'istruzione indicata; in caso contrario si procede in sequenza.

BCS : Esegue il salto se il carry è uguale a 1 (Branch on Carry Set). Se il bit C = 1, si salta all'istruzione indicata; in caso contrario si procede in sequenza.

BEQ : Esegue il salto se il risultato = 0 (Branch on EQual)
Se il bit Z = 1 (vale a dire se l'ultimo risultato è 0 o se l'ultima comparazione è risultata uguale), si passa all'istruzione specificata; in caso contrario si procede in sequenza.

BIT : Test del bit (BIt Test) $Z \leftarrow \sum_i \overline{A_i \wedge M_i}$, $N \leftarrow M7$, $V \leftarrow M6$.

Esegue l'<u>AND</u> virtuale dell'accumulatore e della memoria specificata e posiziona Z di conseguenza. Inoltre i bit 7 e 6 della memoria sono copiati rispettivamente in N e V.

BMI : Esegue il salto se negativo (Branch on MInus).
Se il bit N = 1, si salta all'istruzione indicata; in caso contrario si procede in sequenza.

BNE : Esegue il salto se non uguale a 0 (Branch on Not Equal).
Se il bit Z = 0 (vale a dire se l'ultimo risultato è diverso da 0 o se l'ultima comparazione è risultata diversa), si salta all'istruzione indicata; in caso contrario si procede in sequenza.

BPL : Esegue il salto se positivo o nullo (Branch on Not Equal)
Se il bit N = 0, si salta all'istruzione indicata; in caso contrario si procede in sequenza.

BRK : Interrupt software (BReaK)
Pone il bit B a 1 e simula un'interruzione (salta all'indirizzo contenuto in FFFE, FFFF).

BVC : Esegue il salto se overflow è 0 (Branch on oVerflow Clear).
Se il bit V = 0, si salta all'istruzione indicata; in caso contrario si procede in sequenza.

BVS : Esegue il salto se overflow è 1 (Branch on oVerflow Set).
Se il bit V = 1, si salta all'istruzione indicata; in caso contrario si procede in sequenza.

CLC : Annulla il carry (CLear Carry).
Forza a 0 il bit di carry.

CLD : Annulla il modo decimale (CLear Decimal mode).
Forza a 0 il bit D per impostare il modo binario.

CLI : Autorizza le interruzioni (CLear Interrupt inhibit flag).
Forza a 0 il bit I d'inibizione delle interruzioni.

CLV : Annulla l'indicatore di overflow (CLear oVerflow flag).
Forza a 0 il bit V.

CMP : Comparazione con l'accumulatore (CoMPare accumulator) A - M.
Effettua la sottrazione virtuale registro A - memoria e posiziona, conseguentemente, gli indicatori N, Z e C.

CPX : Comparazione con X (ComPare with X) X - M.
Effettua la sottrazione virtuale registra X - memoria e posiziona, conseguentemente, gli indicatori N, Z e C.

CPY : Comparazione con Y (ComPare with Y) Y - M.
Effettua la sottrazione virtuale registra Y - memoria e posiziona, conseguentemente, gli indicatori N, Z e C.

DEC : Decremento in memoria (DECcrement memory) M ← M - 1.
Diminuisce di 1 il contenuto della memoria indicata. Agisce su N e Z.

DEX : Decrementa X (DEcrement X) X ← X - 1.
Diminuisce di 1 il contenuto del registro X. Agisce su N e Z.

DEY : Decrementa Y (DEcrement Y) Y ← Y - 1.
Diminuisce di 1 il contenuto del registro Y. Agisce su N e Z.

SET DI ISTRUZIONE DEL 6502

EOR : OR esclusivo (Exclusive OR).
Effettua l'OR esclusivo tra l'accumulatore e la memoria specificata. Agisce su N e Z.

INC : Incremento della memoria (INCrement memory).
Aumenta di 1 il contenuto della memoria indicata. Agisce su N e su Z.

INX : Incremento del registro X (INCrement X).
Aumenta di 1 il contenuto del registro X. Agisce su N e Z.

INY : Incremento del registro Y (INCrement Y).
Aumenta di 1 il contenuto del registro Y. Agisce su N e Z.

JMP : Salto incondizionato (JuMP) PC ← Indirizzo
Salta all'indirizzo specificato.

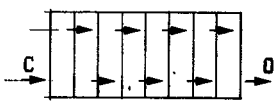
JSR : Chiamata ad un sottoprogramma (Jump to Sub-Routine) PC ↓ ;
PC ← Ad.
Salva il PC (Program Counter) nella pila (indirizzo di ritorno) poi salta all'indirizzo specificato.

LDA : Carica l'accumulatore (LoaD Accumulator) A ← M.
Pone nell'accumulatore il contenuto della memoria specificata. Agisce su N e Z.

LDX : Carica il registro X (LoaD X register) X ← M.
Pone nel registro X il contenuto della memoria specificata. Agisce su N e Z.

LDY : Carica il registro Y (LoaD Y register) Y ← M.
Pone nel registro Y il contenuto della memoria specificata. Agisce su N e Z.

LSR : Spostamento a destra (Logical Shift Right)
Scala di un posto a destra l'accumulatore o la memoria specificata. Agisce su N, Z e C.



NOP : Nessuna operazione (No OPeration) PC ← PC + 1.
Istruzione nulla. L'esecuzione richiede 2 cicli macchina.

ORA : OR logico (OR Accumulator) A ← A∨M.
Effettua l'<u>OR</u> logico tra l'accumulatore e la memoria specificata
Agisce su N e Z.

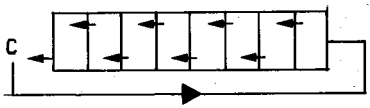
PHA : Impila A (PusH Accumulator) A ↓ : (S) ← A ; S ← S - 1.
Pone l'accumulatore in cima alla pila e il pointer viene decrementato di uno.

PHP : Impila P (PusH Processor status register).
P ↓ : (S) ← P ; S ← S - 1.
Pone il registro di stato P in cima alla pila e decrementa il puntatore di uno.

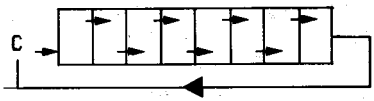
PLA : Disimpila verso A (PuLl Accumulator) A↑: S ← S + 1 ;
A <- (S).
Trasferisce verso l'accumulatore il contenuto della cima della pila e riaggiorna il puntatore. Ripristina N e Z.

PLP : Disimpila verso P (PuLl P register) P : S ← S + 1 ; P ← (S).
Trasferisce verso il registro di stato P il contenuto della cima della pila e riaggiorna il puntatore. Ripristina tutti gli indicatori.

ROL : Rotazione a sinistra (ROtate Left)
Esegue la rotazione verso sinistra dell'accumulatore o di una memoria. Il vecchio valore del bit di riporto rientra a destra mentre il bit che esce alla sinistra diventa il nuovo valore di C. Ripristina N, Z e C.



ROR : Rotazione a destra (ROtate Right)
Esegue la rotazione verso destra dell'accumulatore o di una memoria. Il vecchio valore del bit di riporto rientra a sinistra mentre il bit che esce alla destra va a rimpiazzare C. Ripristina N, Z e C.



RTI : Ritorno dall'interruzione (ReTuen from Interrupt).
P ↑ ; PC ↑
Ritorno da una routine di interrupt: recupera sulla pila PC e P che sono stati salvati dal meccanismo d'interruzione.

RTS : Ritorno da un sottoprogramma (ReTurn from Subroutine) PC↑ .
Recupera sulla pila PC che era stato salvato dall'ultima JSR.

SBC : Sottrazione con riporto (SuBtract with Carry).
A ← A - M - C
Si sottrae all'accumulatore la memoria specificata oltre al bit negativo di riporto. Opera in modo binario o decimale. Agisce su N, V, Z, C.

SEC : Pone a 1 il carry (SEt Carry flag).
Forza a 1 il bit C.

SED : Pone in modo decimale (SEt Decimal mode).
Forza a 1 il bit D.

SET DI ISTRUZIONE DEL 6502

SEI : Inibisce le interruzioni (SEt Interrupt inhibit flag).
Forza a 1 il bit I.

STA : Scrive l'accumulatore (STore Accumulator) M ← A.
Trasferisce il contenuto dell'accumulatore nella memoria specificata.

STX : Scrive il registro X (STore X register) M ← X.
Trasferisce il contenuto del registro X nella memoria specificata.

STY : Scrive il registro Y (STore Y register) M ← Y.
Trasferisce il contenuto del registro Y nella memoria specificata.

TAX : Trasferimento di A in X. X ← A. Agisce su N e Z.

TAY : Trasferimento di A in Y. Y ← A. Agisce su N e Z.

TSX : Trasferimento di S in X. X ← S. Agisce su N e Z.

TXA : Trasferimento di A in S. A ← S. Agisce su N e Z.

TXS : Trasferimento di X in S. S ← X. Non agisce sugli indicatori.

TYA : Trasferimento di Y in A. A ← Y. Agisce su N e Z.

Tabella di decodifica

Questa tabella rappresenta il contrario rispetto a quella che segue. In funzione del codice esadecimale AB, fornisce il codice mnemonico e il modo di indirizzamento corrispondente. Esepio: A9 → LDA IMM (linea A, colonna 9). Nessun indirizzamento = indiretto o relativo.

A \ B	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	BRK	ORA IND,X				ORA PZ,X	ASL PZ,X		PHP	ORA IMM	ASL A			ORA ABS	ASL ABS	
1	BPL	ORA IND,Y				ORA PZ,X	ASL PZ,X		CLC	ORA ABS,Y				ORA ABS,X	ASL ABS,X	
2	JSR	AND IND,X			BIT PGE Z	AND PGE Z	ROL PGE Z		PLP	AND IMM	ROL A		BIT ABS	AND ABS	ROL ABS	
3	BMI	AND IND,Y				AND PZ,X	ROL PZ,X		SEC	AND ABS,Y				AND ABS,X	ROL ABS,X	
4	RTI	EOR IND,X				EOR PGE Z	LSR PGE,Z		PHA	EOR IMM	LSR A		JMP ABS	EOR ABS	LSR ABS	
5	BVC	EOR IND,Y				EOR PZ,X	LSR PZ,X		CLI	EOR ABS,Y				EOR ABS,X	LSR ABS,X	
6	RTS	ADC IND,X				ADC PGE Z	ROR PGE Z		PLA	ADC IMM	ROR A		JMP IND	ADC ABS	ROR ABS	
7	BVS	ADC IND,Y				ADC PZ,X	ROR PZ,X		SEI	ADC ABS,Y				ADC ABS,X	ROR ABS,X	
8		STA IND,X			STY PGE Z	STA PGE Z	STX PGE Z		DEY		TXA		STY ABS	STA ABS	STX ABS	
9	BCC	STA IND,Y			STY PZ,X	STA PZ,X	STX PZ,Y		TYA	STA ABS,Y	TXS			STA ABS,X		
A	LDY IMM	LDA IND,X	LDX IMM		LDY PGE Z	LDA PGE Z	LDX PGE Z		TAY	LDA IMM	TAX		LDY ABS	LDA ABS	LDX ABS	
B	BCS	LDA IND,Y			LDY PZ,X	LDA PZ,X	LDX PZ,Y		CLV	LDA ABS,Y	TSX		LDY ABS,X	LDA ABS,X	LDX ABS,Y	
C	CPY IMM	CMP IND,X			CPY PGE Z	CMP PGE Z	DEC PGE Z		INY	CMP IMM	DEX		CPY ABS	CMP ABS	DEC ABS	
D	BNE	CMP IND,Y				CMP PZ,X	DEC PZ,X		CLD	CMP ABS,Y				CMP ABS,X	DEC ABS,X	
E	CPX IMM	SBC IND,X			CPX PGE Z	SBC PGE Z	INC PGE Z		INX	SBC IMM	NOP		CPX ABS	SBC ABS	INC ABS	
F	BEQ	SBC IND,Y				SBC PZ,X	INC PZ,X		SED	SBC ABS,Y				SBC ABS,X	INC ABS,X	

SET DI ISTRUZIONE DEL 6502

Codici operativi / numero dei cicli macchina (n) / numero dei byte occupati (#)

INSTRUCTIONS		IMMEDIAT			ABSOLU			PAGE ZERO			ACCUM.			INHERENT			(IND,X)			(IND),Y			PAGE Z,X			ABS,X			ABS,Y			RELATIF			INDIRECT			PAGE Z,Y			REGISTRE D'ETAT	MNEMO.
MNEMO	EFFET	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	7 6 5 4 3 2 1 0 N V . B D I Z C	
A D C	A+M+C → A (1) (4)	69	2	2	6D	4	3	65	3	2							61	6	2	71	5	2	75	4	2	7D	4	3	79	4	3										N V . . . . Z C	A D C
A N D	A∧M → A (1)	29	2	2	2D	4	3	25	3	2							21	6	2	31	5	2	35	4	2	3D	4	3	39	4	3										N . . . . . Z .	A N D
A S L	C ← [7 … 0] ← 0				0E	6	3	06	5	2	0A	2	1										16	6	2	1E	7	3													N . . . . . Z C	A S L
B C C	BRANCH.SI C=0 (2)																															90	2	2								. . . . .	B C C
B C S	BRANCH.SI C=1 (2)																															B0	2	2								. . . . .	B C S
B E Q	BRANCH.SI Z+1 (2)				2C	4	3	24	3	2																						F0	2	2								. . . . .	B E Q
B I T	AAM																																									M7 M6 . . . . Z .	B I T
B M I	BRANCH.SI N=1 (2)																															30	2	2								. . . . .	B M I
B N E	BRANCH.SI Z=0 (2)																															D0	2	2								. . . . .	B N E
B P L	BRANCH.SI N=0 (2)																															10	2	2								. . . . .	B P L
B R K	BREAK													00	7	1																										. . . 1 . 1 . .	B R K
B V C	BRANCH.SI V=0 (2)																															50	2	2								. . . . .	B V C
B V S	BRANCH.SI V=1 (2)																															70	2	2								. . . . .	B V S
C L C	0 → C													18	2	1																										 0	C L C
C L D	0 → D													D8	2	1																										. . . . 0 . . .	C L D
C L I	0 → I													58	2	1																										. . . . . 0 . .	C L I
C L V	0 → V													B8	2	1																										. 0	C L V
C M P	A - M	C9	2	2	CD	4	3	C5	3	2							C1	6	2	D1	5	2	D5	4	2	DD	4	3	D9	4	3										N . . . . . Z C	C M P
C P X	X - M	E0	2	2	EC	4	3	E4	3	2																																N . . . . . Z C	C P X
C P Y	Y - M	C0	2	2	CC	4	3	C4	3	2																																N . . . . . Z C	C P Y
D E C	M - 1 → M				CE	6	3	C6	5	2													D6	6	2	DE	7	3													N . . . . . Z .	D E C
D E X	X - 1 → M													CA	2	1																										N . . . . . Z .	D E X
D E Y	Y - 1 → Y													88	2	1																										N . . . . . Z .	D E Y
E O R	A ● M → A (1)	49	2	2	4D	4	3	45	3	2							41	6	2	51	5	2	55	4	2	5D	4	3	59	4	3										N . . . . . Z .	E O R
I N C	M+1 M				EE	6	3	E6	5	2													F6	6	2	FE	7	3													N . . . . . Z .	I N C
I N X	X+1 → X													E8	2	1																										N . . . . . Z .	I N X
I N Y	Y+1 → Y													C8	2	1																										N . . . . . Z .	I N Y
J M P	SAUT INCOND				4C	3	3																												6C	5	3					. . . . .	J M P
J S R	APPEL S/P				20	6	3																																			. . . . .	J S R
L D A	M → A (1)	A9	2	2	AD	4	3	A5	3	2							A1	6	2	B1	5	2	B5	4	2	BD	4	3	B9	4	3										N . . . . . Z .	L D A

SET DI ISTRUZIONE DEL 6502

Codici operativi / numero dei cicli macchina (n) / numero dei byte occupati (#)

MNEMO	INSTRUCTIONS EFFET	IMMEDIAT OP	IMMEDIAT n	IMMEDIAT #	ABSOLU OP	ABSOLU n	ABSOLU #	PAGE ZERO OP	PAGE ZERO n	PAGE ZERO #	ACCUM OP	ACCUM n	ACCUM #	INHERENT OP	INHERENT n	INHERENT #	(IND,X) OP	(IND,X) n	(IND,X) #	(IND,Y) OP	(IND,Y) n	(IND,Y) #	PAGE Z,X OP	PAGE Z,X n	PAGE Z,X #	ABS,X OP	ABS,X n	ABS,X #	ABS,Y OP	ABS,Y n	ABS,Y #	RELATIF OP	RELATIF n	RELATIF #	INDIRECT OP	INDIRECT n	INDIRECT #	PAGE Z,Y OP	PAGE Z,Y n	PAGE Z,Y #	REGISTRE D'ETAT 7 6 5 4 3 2 1 0 N V . B D I Z C	MNEMO
L D X	M → X (1)	A2	2	2	AE	4	3	A6	3	2																		BE	4	3							B6	4	2	N . . . . . Z .	L D X
L D Y	M → Y (1)	A0	2	2	AC	4	3	A4	3	2																														N . . . . . Z .	L D Y
L S R	0 → [7 … 0] → C				4E	6	3	46	5	2	4A	2	1										56	6	2	5E	7	3												0 . . . . . Z C	L S R
N O P	PAS D'OPERATION													EA	2	1																								. . . . .	N O P
O R A	AVM → A	09	2	2	0D	4	3	05	3	2							01	6	2	11	5	2	15	4	2	1D	4	3	19	4	3									N . . . . . Z .	O R A
P H A	A → Ms S-1 → S													48	3	1																								. . . . .	P H A
P H P	P → Ms S-1 → S													08	3	1																								. . . . .	P H P
P L A	S+1 → S Ms → A													68	4	1																								N . . . . . Z .	P L A
P L P	S+1 → S Ms → P													28	4	1																								(RESTAURES)	P L P
R O L	[7 … 0] ← [C]				2E	6	3	26	5	2	2A	2	1										36	6	2	3E	7	3												N . . . . . Z C	R O L
R O R	[C] → [7 … 0]				6E	6	3	66	5	2	6A	2	1										76	6	2	7E	7	3												N . . . . . Z C	R O R
R T I	RETOUR INTER													40	6	1																								(RESTAURES)	R T I
R T S	RETOUR S/P													60	6	1																								. . . . .	R T S
S B C	A - M - C → A (1) (4)	E9	2	2	ED	4	3	E5	3	2							E1	6	2	F1	5	2	F5	4	2	FD	4	3	F9	4	3									N V . . . . Z (3)	S B C
S E C	1 → C													38	2	1																								 1	S E C
S E D	1 → D													F8	2	1																								. . . . 1 . . .	S E D
S E I	1 → D													78	2	1																								. . . . . 1 . .	S E I
S T A	A → M				8D	4	3	85	3	2							A1	6	2	91	6	2	95	4	2	9D	5	3	99	5	3									. . . . .	S T A
S T X	X → M				8E	4	3	86	3	2																														. . . . .	S T X
S T Y	Y → M				8C	4	3	84	3	2													94	4	2													96	4	2	. . . . .	S T Y
T A X	1 → X													AA	2	1																								N . . . . . Z .	T A X
T A Y	A → Y													A8	2	1																								N . . . . . Z .	T A Y
T S X	S → X													BA	2	1																								N . . . . . Z .	T S X
T X A	X → A													8A	2	1																								N . . . . . Z .	T X A
T X S	X → S													9A	2	1																								. . . . .	T X S
T Y A	Y → A													98	2	1																								N . . . . . Z .	T Y A

(1) Ajouter 1 à "n" si changement de page

(2) Ajouter 1 à "n" si branchement dans une page différente

(3) Emprunt = $\overline{\text{retenue}}$

(4) En mode décimal, l'indicateur Z est inopérant. L'accumulateur doit être testé pour savoir si le résultat est nul.

X	Index X	+	Addition
Y	Index Y	-	Soustraction
A	Accumulateur	^	Et
M	Mémoire	v	Ou
Ms	Mémoire (pile)	●	Ou exclusif
M_7	Mémoire (bit 7)	n	Nombre de cycles
M_6	Mémoire (bit 6)	#	Nombre d'octets

1 - Posizionamento del cursore

Mantenere il cursore accanto alla domanda fino a quando la risposta non sia quella corretta.

La posizione del cursore è **memorizzata** prima della sua scomparsa e ripristinata per una eventuale nuova impostazione errata.

Se la risposta è errata, viene **visualizzata**.

Esempio:

```
 5 HOME
10 PRINT "RISPONDERE CON S(I O N(O ";: GOSUB 100
20 INPUT "";R$
30 IF R$ = "S" THEN HTAB PH: VTAB PV: PRINT "SI"
35 IF R$ = "N" THEN HTAB PH: VTAB PV: PRINT "NO"
40 IF LEFT$ (R$,1) < > "S" AND LEFT$ (R$,1) < > "
   N" THEN HTAB PH: VTAB PV: CALL -868: GOTO 20
50 PRINT "FINE": PRINT : GOTO 10
```

```
100 PH =  PEEK (36) + 1
200 PV = PEEK (37) + 1
210 RETURN
```

PH è la posizione orizzontale (colonna)
PV è la posizione verticale (riga)

CALL -868 visualizza il resto della linea dopo il punto dove si trovava il cursore.

2 - Simulare INPUT X$

La stringa letta è inscritta da $200 (512) a $2FF (768)

APPLESOFT	Assemblatore
CALL-10964	JSR $D52C

Tutti i caratteri sono accettati fino al valore 255, ma CTRL-X annulla la linea e RETURN convalida l'input.

3 - Come impedire il list di un programma

POKE 2049,0 : POKE 2050,0

pone a zero il puntatore d'inizio relativo alla seconda linea di istruzioni.

Per ritrovare il suo esatto valore, cercare il primo byte 00 che indica la fine della prima linea di istruzioni e porre a 1 il suo valore.

Esempio:

```
  LIST

 10  REM  COME N.3
 20  PRINT : END

  POKE2049,0:POKE2050,0

  LIST


  CALL -151

 *800.81F                    14 Ø8

 0800- 00 00 00 0A 00 B2 20 43
 0808- 4F 4D 4D 45 4E 54 20 4E
 0810- 30 20 33 00 1C 08 14 00
 0818- BA 3A 80 00 00 00 0A 00
```

I comandi NEW e FP annullano anche questo puntatore senza cancellare il programma.

4 - Caricare un programma dopo l'altro

a) Caricare il primo programma in memoria RAM

b) Modificare il puntatore di inizio programma per il punto dopo il byte 00 dell'ultima linea di istruzione del primo programma.

c) Caricare il programma che segue il memoria RAM.

d) Modificare il puntatore di inizio programma per il punto di inizio relativo al primo programma.

TEXTTAB puntatore di inizio programma $67,$68
PRGEND puntatore di fine programma $AF,$B0

```
NEW

100 REM PG CODA

SAVE PCODA

REM PG TESTA                    (a)

CALL -151

*AF.B0

00AF- 15
00B0- 08
*800.815

0800- 00 12 08 0A 00 B2 20 50
0808- 47 20 44 45 20 54 45 54
0810- 45 00 00 00 64 0A
*67:12 08                       (b)

*3D0G

 LOAD PCCODA                    (c)
 CALL -151

*67:01 08                       (d)

*3D0G

 LIST

10 REM  PG TESTA
100 REM  PG CODA
```

5 - Come impedire l'accessa da tastiera

Oltre alla neutralizzazione del tasto RESET è necessario prevenire anche l'uso accidentale del tasto CTRL-C che provoca l'interruzione del programma in corso con il messaggio:

BREAK IN n. della linea di istruzione dove il programma è stato fermato.

La soluzione proposta utilizza il **trattamento d'errore:** CTRL-C corrisponde al codice d'errore n. 255 e, quando è riscontrato, si annulla con l'istruzione RESUME:

```
1 ON ERR GOTO 1000
1000 IF PEEK(222) = 255 THEN RESUME
```

6 - Tutti i comandi sono interpretati come RUN

POKE 214,128

Un valore superiore o uguale a 128 nell'indirizzo 214 in $D6 ha un effetto irreversibile su tutti i comandi o istruzioni BASIC, tutto è trasformato in RUN. Sono immuni solo i comandi d'accesso a programmi su disco.

Fare PR#6 per rinizializzare il sistema.

7 - Inibizione del tasto RESET

L'effetto del RESET sul sistema dipende dal contenuto delle locazioni di memoria $3F2 e $3F3.

L'indirizzo contenuto in queste locazioni è quello verso il quale il sistema "salta" se viene premuto il tasto RESET.

indirizzo		Valore di default		
Dec	Hex	Dec	Hex	
1010	3F2	191	BF	ritorno al BASIC sotto DOS (arresto del programma)
1011	3F3	157	9D	
1012	3F4	56	38	OR escl. di (1011) e $A5

Il byte d'indirizzo $3F4 dev'essere ricalcolato per modificare l'effetto di RESET.
Il suo valore è ottenuto con CALL-1169 ($FB6F) poi PRINT POKE (1012).

a) inibizione: (il programma in corso non si arresta con RESET)

```
* 3F2 : 00 03 A6

* 300 : 20 EA 03      JSR $03EA (DOS)
* 303 : 20 98 D8      JSR $D898 (CONT)
* 305 : 4C D2 D7      JMP $D7D2 (NEWSTT)
```

b) inibizione di tutto il sistema (dopo aver premuto RESET)

```
* 300 : 4C 00 03
```

Il rilancio di tutto il sistema è possibile solo dopo l'interruzione.

c) il tasto RESET rilancia il sistema come quando lo si accende:

basta fare POKE 1012,0

d) disinibizione * 3F2 : BF 9D 38

8 - Attesa di un carattere da tastiera

a) 10 X = PEEK(-16384) : IF X < 128 THEN 10
   20 POKE -16368,0 : X$ = CHR$(X-128)

b) 10 WAIT -16384,128 : X = PEEK(-16384)-128 : POKE -16368,0

c) 10 GET X$

d) 10 CALL -756 (RDKEY)

9 - Modificare la visualizzazione del list di un programma BASIC

- POKE 33,33

La finestra di schermo è ridotta a 33 colonne di larghezza.
Il comando LIST visualizza le linee di istruzione senza margine.

- TEXT annulla il comando precedente.

- Il carattere : consente di introdurre l'indentazione delle linee di istruzione.

- POKE 33,28
  facilita l'incolonnamento dei REM: la disposizione, alla memorizzazione, non è modificata dal comando LIST.

- TEXT oppure POKE 33,40 riportano al modo standard.

10 - GOTO calcolato utilizzando &

Si scrive l'espressione &.

Il sottoprogramma è memorizzato a partire dalla locazione $300. Dunque gli indirizzi $3F5, $3F6 e $3F7 devono essere preventivamente registrati con l'istruzione JMP $300 per rendere possibile il salto dell'interprete quando è incontrata l'espressione &.

    * 3F5 : 4C 00 03

oppure:

    POKE 1013,76 : POKE 1014,0 : POKE 1015,3

I "COME...?"

Sottoprogramma di valutazione dell'espressione e salto alla linea calcolata.

```
 * 300L

 0300-    20 7B DD      JSR    $DD7B
 0303-    20 52 E7      JSR    $E752
 0306-    20 1A D6      JSR    $D61A
 0369-    90 03         BCC    $030E
 030B-    4C 41 D9      JMP    $D941
 030E-    A2 5A         LDX    #$5A
 0310-    4C 12 D4      JMP    $D412
```

$DD7B	FRMEVL	valutazione dell'espressione; il risultato è posto in FAC.
$E752	GETADR	conversione di FAC in valore intero; il risultato è posto in $50, $51
$D61A	FNDLIN	cerca se la linea calcolata fa parte del programma
$D941	GOTO+	salta alla linea trovata
$D412	ERROR	eventuale errore con codice $5A = 90 UNDEF'D STATEMENT

11 - Stampa con D decimali

a) DEF FNF(X) = INT(X*10^D)/10^D

invece di stampare X, si stamperà FNF(X)

Nota: l'istruzione PRINT di un numero reale non visualizza gli zeri più a destra della parte frazionaria e neppure quelli più a sinistra della parte intera.

Se $0.01 \leqslant |X| < 999\ 999\ 999.2$ il numero è in virgola fissa, altrimenti è sotto forma mantissa, esponente

sx.xxx xxx xxEstt

s è il segno
E significa potenza di 10
x e t sono cifre tra 0 e 9

b) arrotondare a D decimali

La funzione INT(X) da come risultato il più piccolo intero inferiore a X, questo pone qualche problema se X è negativo.

```
Così  ?INT(-5.3)
      -6
```

Dunque è necessario tener conto del segno di X nell'arrotondamento.

```
Così   ?INT(ABS(-5.3))/SGN(-5.3)
       -5

DEF FN AR(X) = INT(ABS(X)*10^D+.5)/10^D*SGN(X)

Fare   PRINT FNAR(X)
```

c) Troncare a D decimali con notazione flottante

```
10 X$ = STR$(X)
20 FOR I = 1 TO LEN(X$):IF MID$(X$,I,1)<>"E" THEN NEXT I
30 FOR J = 1 TO I-1:IF MID$(X$,J,1)<>"." THEN NEXT J
40 IF J+D < = I-1 THEN N=J+D:GOTO 60
50 N=I-1
60 PRINT LEFT$(X$,N)+MID$(X$,I)
```

12 - Incolonnamento a destra in una zona di C caratteri

```
A$ = STR$(FN AR(X))
C$ = "STR$(FN AR(X))-" : REM C caratteri "spazio"
PRINT RIGHT$(C$+A$,C)
```

13 - Conoscere l'indirizzo di una variabile

Bisogna distinguere le variabili numeriche semplici dalle variabili alfanumeriche.
Dopo i due byte che rappresentano i primi due caratteri del **nome**, il sistema riserva 5 byte per conservare il **valore** reale o intero di una variabile numerica.

Esempio: A = 1

41	00	81	00	00	00	00
NOME		VALORE				

B% = 32767

C2	80	7F	FF	00	00	00
NOME		VALORE				

Il caso delle variabili alfanumeriche, dove il valore è una stringa di caratteri, è differente poichè nei 5 byte che seguono il nome della

variabile, si trova la **lunghezza** della stringa e l'**indirizzo di inizio** della stringa.

Esempio: A$ = ""

41	80	00			00	00

NOME — LUNGHEZZA — PUNTATORE

L'indirizzo cercato è quello del valore di una variabile **numerica** dunque il puntatore del valore.

L'indirizzo di una stringa di caratteri è quello contenuto nei byte +1 e +2 in rapporto al puntatore del valore.

Questo puntatore del valore (VARPNT) è memorizzato in $83, $84 (131, 132) e contiene l'indirizzo dell'ultima variabile trattata dall'Applesoft.

n Applesoft si scrive:

```
10 X=A : REM    si ricerca dell'indirizzo A
20 A=PEEK(131) + 256*PEEK(132)
30 PRINT A
40 A=X : REM si ristabilisce il valore di A
```

In linguaggio macchina, ci si può servire della routine PTRGET d'indirizzo $DFE3 per recuperare nell'accumulatore e nel registro Y, il byte meno significativo e quello più significativo relativo al puntatore della variabile della quale si cerca l'indirizzo.

Grazie all'operatore & seguito dal nome della variabile, si entra in un sottoprogramma in linguaggio macchina che rinvia agli indirizzi 778 ($30A) e 779 ($30B) il valore cercato.

```
*300L

0300-   20 E3 DF    JSR   $DFE3
0303-   8D 0A 03    STA   $030A
0306-   8C 0B 03    STY   $030B
0309-   60          RTS
```

In una tabella di valori numerici interi, il valore occupa solo due byte per ciascuna variabile indice. Il puntatore del valore sarà

utilizzato direttamente; l'indirizzo della variabile indice contiene il byte più significativo, seguito dal byte meno significativo.

```
                    1  POKE 1013,76: POKE 1014,0: POKE 1015,3
                    2 X = 0:L = 0:P = 0:A$ = "APPLESOFT"
                    5  DIM A%(100):A%(1) = 32767
                    10  & A%(1)
                    30 X = PEEK (778) +  PEEK (779) * 256
                    40  PRINT X
                    45  PRINT 256 * PEEK (X) +  PEEK (X + 1)
                    50 A$ = LEFT$ (A$,5)
                    60  & A$
                    70 X = PEEK (778) +  PEEK (779) * 256
Lunghezza           80 L = PEEK (X)
puntatore           90 P = PEEK (X + 1) +  PEEK (X + 2) * 256
                    100  PRINT L,P;" ";
                    110  FOR X = P TO P + L - 1
Stringa             120  PRINT  CHR$ ( PEEK (X));: NEXT X

                     RUN
Indirizzo           2365
                    32767
Valore              5                    38395 APPLE
```

14 - Listato su stampante

Se la scheda di interfaccia della stampante è posta nel connettore n.1:

```
PR#1
LIST
```

Se le linee di istruzione superano i 30 caratteri, è necessario modificare il numero dei caratteri editati per linea sulla stampante per evitare il formato classico (immagine di schermo).

```
PR#1
PRINT "CTRL-I 80 N" per 80 caratteri per linea.
```

Questa istruzione rende impossibile la visualizzazione sullo schermo. Per uscire da tale stato battere RESET oppure

```
PR#0
```

15 - Cambiare la pagina di schermo

```
POKE -16299,0  visualizza la pagina n.1                    *C055
POKE -16300,0  visualizza la pagina n.2                    *C054
```

I "COME...?"

```
POKE -16304,0  visualizza in modo grafico                    *C050
POKE -16303,0  visualizza in modo testo                      *C051

POKE -16297,0  visualizza in modo HI-RES senza cancellare    *C057
POKE -16298,0  visualizza in modo GR senza cancellare        *C056

POKE -16302,0  grafica a tutto schermo                       *C053
POKE -16301,0  grafica con 4 linee di testo                  *C052
```

16 - Modifica della finestra di schermo

TEXT regola la finestra per il suo massimo valore:

```
     Larghezza          : $21(33) : WNDWDTH = $28(40)
     Margine sinistro   : $20(32) : WNDLFT  = $0(0)
     Margine superiore  : $22(34) : WNDTOP  = $0(0)
     Margine inferiore  : $23(35) : WNDBTM  = $18(24)

POKE 33,larghezza          compreso tra 1 e 40
POKE 32,margine sinistro   margine sinistro + larghezza inferiore a 39
POKE 34,margine superiore  compreso tra 0 e 23
POKE 35,margine inferiore  più alto del margine superiore e inferiore
                           a 24.
```

Il margine sinistro si riposiziona con un RETURN (PRINT).

17 - Per far si che PRINT stampi caratteri minuscoli

PR#1 (pone in linea la stampante)

```
 POKE 243,32:PRINT"10 BUONGIORNO":NORNAL
10 buongiorno
```

L'indirizzo 243 o $F3 detto anche ORMASK serve nell'Applesoft per forzare il modo FLASH a gruppi di cifre ($60-$7F) per l'istruzione OR ORMASK.

```
con ORMASK = $40 in modo FLASH
           = $0  in modo NORMAL e INVERSE
```

Le minuscole differiscono dalle maiuscole per il bit 5 del loro codice che vale 1 per le minuscole ($E0-$FF) e 0 per le maiuscole ($C0-$DF).

Per una operazione OR ORMASK con OR MASK = $20(32), il bit 5 è posto a 1 dalla POKE 243,32 per stampare caratteri minuscoli.

18 - Cancellazione dello schermo

HOME oppure
o ESC @ } CALL-936 o, ancora, FC58 G cancella tutto il testo dallo schermo e posiziona il cursore in alto a destra.

ESC E oppure CALL-868 o, ancora, FC9C G cancella dopo la posizione del cursore fino alla fine della linea corrente (o fino al bordo destro della finestra di schermo).

ESC F oppure CALL-958 o, ancora, FC42 G cancella dopo la posizione del cursore fino alla fine dello schermo (o della finestra).

19 - Scrivere sullo schermo dal basso verso l'alto

CALL -998 : CALL -998 inserito in ciascuna istruzione PRINT.

20 - Scrivere sullo schermo da destra a sinistra

CALL -1008 : CALL -1008 inserito in ciascuna istruzione PRINT X$.

21 - Spostare tutto il testo verso l'alto

CALL -912 oppure FC70 G

Questa operazione è realizzata **automaticamente** quando il cursore raggiunge la venticinquesima linea che apparirà alla ventiquattresima con lo scalare di una linea di tutto il testo verso l'alto.

22 - Proteggere un INPUT con un valore di default

Il valore di default ha lunghezza di un carattere

```
 10  REM  PROTEZIONE PER DEFAULT
 20 DE$ = "O": REM   VALORE PER DAFAULT
 30  PRINT "DOMANDA ? ";DE$;
 40 PH =  PEEK (36):PU =  PEEK (37) + 1
 41  IF PV > 23 THEN PV = 23: REM  ATTENZIONE ALLO SCROLL
 42  CALL - 1008: REM  RECUPERO DI UNA POSIZIONE
 50  INPUT "";RE$
 60  IF RE$ = "" THEN RE$ = DE$
 70  HTAB PH: VTAB PV: PRINT RE$
```

Quando l'INPUT è eseguita si vedrà il cursore lampeggiare sul valore di default. Se come risposta si batte RETURN il valore accettato sarà quello di default. Se si desidera impostare un altro valore, si batta tale valore su quello presentato, poi battere RETURN.

23 - Prevedere la lunghezza di un programma

In generale un programma occupa tanti byte quanti sono i caratteri che lo compongono poichè esso è memorizzato tale quale, fatte salve le parole riservate che sono memorizzate con un codice di un byte.

Per ciò che concerne le variabili, ciascuna variabile numerica semplice reale o intera occupa 7 byte, ciascuna stringa occupa (7 + lunghezza) byte.

Una matrice occupa x(n+1) + 2d + 3
n è la dimensione della matrice (compreso lo 0)
x = 5 (numero reale)
x = 2 (intero)
x = 3 (stringa di caratteri)
d numero di dimensioni

Si guadagna spazio in memoria sopprimendo tutti gli spazi inutili, ponendo più istruzioni sulla stessa linea, evitare i REM, utilizzare variabili invece di costanti.
Utilizzare i GOSUB ai quali fare appello più volte in sequenze d'istruzioni simili.

24 - Fare musica con Apple

Un motivo è definito da un elenco relativo alla coppia I, J
I è l'altezza della nota e la sua frequenza
J è la durata di questa nota

Un programma, scritto in linguaggio macchina, consente di attivare l'altoparlante con LDA $C030 ad intervalli regolari:
- Il registro X è inizializzato con il valore 1, un 'bip', e diminuisce fino a 0, fino al successivo 'bip'; più I è basso, più la frequenza, dunque l'altezza, è elevata.
- Diminuisce anche il registro Y e il suo passaggio a zero fa diminuire J, la durata, la quale, avvicinandosi allo zero provoca la fine dell'esecuzione di una nota.

GAMMA : (con il programma seguente). I valori di I:
255,242,230,216,204,192,182,172,162,152,144,136,128,
128,121,115,108,102, 96, 91, 86, 81, 76, 72, 68, 64,
64, 60, 57, 54, 51, 48, 45, 43, 40, 38, 36, 34, 32
SOL SOL LA LA SI DO DO RE RE MI FA FA SOL

```
0302-   AD 30 C0   LDA   $C030
0305-   88         DEY
0306-   D0 05      BNE   $030D
0308-   CE 01 03   DEC   $0301
030B-   F0 09      BEQ   $0316
```

```
030D-   CA          DEX
030E-   D0 F5       BNE   $0305
0310-   AE 00 03    LDX   $0300
0313-   4C 02 03    JMP   $0302
0316-   60          RTS
```

Per suonare un motivo, è necessario chiamare il sottoprogramma per ciascuna nota succesiva.

In BASIC, i valori della coppia I, J sono letti da una serie di DATA fino al valore 0,0

Il programma seguente è memorizzato all'inizio del programma con degli statement POKE A,V.

Esempio: due piccoli motivi "sintetici"

```
10  REM  MUSICA
20  POKE 770,173: POKE 771,48: POKE
     772,192: POKE 773,136: POKE
     774,208: POKE 775,5: POKE 77
     6,206: POKE 777,1: POKE 778,
     3: POKE 779,240: POKE 780,9:
      POKE 781,202
30  POKE 782,208: POKE 783,245: POKE
     784,174: POKE 785,5: POKE 78
     6,3: POKE 787,76: POKE 788,2
     : POKE 789,3: POKE 790,96: POKE
     791.0: POKE 792,0
40  READ I,J: IF J = 0 THEN 70
50  POKE 768,I: POKE 769,J: CALL
    770
60  GOTO 40
70  IF F = 1 THEN  END
80 F = 1: INPUT "ANCORA UNA VOLTA ? ";R$
90  GOTO 40
100  DATA  114,120,144,60,114,255
     ,1,120,128,120,144,60,128,12
     0,114,60,144,120,171,255,228
     ,255,0,0
200  DATA  0,160,128,255,152,40,1
     71,80,192,40,228,255,1,40,0,
     160,192,255,192,40,171,80,15
     2,40,128,255,0,0
```

25 - Disegnare, ingrandire far girare una figura

La codifica di una figura può essere semplificata utilizzando un byte per ciascun vettore elementare tracciato:

- dirigere verso l'alto → 4
- dirigere verso destra → 5
- dirigere verso il basso → 6
- dirigere verso sinistra → 7

I vettori successivi sono posti in una zona scelta dall'utilizzatore: si chiama tavola delle figure.
Questa tavola deve contenere:
- nel primo byte il numero della figura (esempio 1)
- nel terzo e nel quarto i dati per reperire l'inizio della prima figura (04 00 per una figura)
- poi i vettori della figura.

Bisogna precisare all'inizio del programma, l'indirizzo di inizio della tavola delle figure. Questo indirizzo è posto in $EB o 232 e $E9 o 233 con la parte bassa dell'indirizzo in $E8 (meno significativo).

In BASIC, la succesione dei vettori del disegno posti in DATA con 0 come fine dell'elenco, è letto e memorizzato a partire dal quinto byte della tavola (se è stata prevista una sola figura).

La figura scelta nell'esempio che segue è un petalo stilizzato.

Un fiore con un petalo come figura definita

```
 LIST

10  HGR
20  HCOLOR= 3
25  REM  LA TAVOLA DELLE FIGURE E'
     ALL'INDIRIZZO $300 O 768
30  POKE 232,0: POKE 233,3
35  REM   UNA SOLA FIGURA ... UN
      PETALO
40  POKE 768,1: POKE 769,0: POKE
     770,4: POKE 771,0
42  RESTORE :T = 0
43  READ D: POKE 772 + T,D: IF D =
     0 THEN 48
45 T = T + 1: GOTO 43
48 X = 140:Y = 80
50  SCALE= 3
52  REM ECCO IL FIORE
55  FOR R = 0 TO 64 STEP 4
58  ROT= R
60  DRAW 1 AT X,Y
70  NEXT R
80  END
100  DATA  4,4,4,5,4,4,4,5,4,4,4,
     5,4,4,4,5,4,5,4,5,4,5,4,5,5,
     5,4,5,5,5,4,5,5,5,4,5,5,5
200  DATA  6,6,6,7,6,6,6,7,6,6,6,
     7,6,6,6,7,6,7,6,7,6,7,6,7,7,
     7,6,7,7,7,6,7,7,7,6,7,7,7 ,0
300  REM LA DIREZIONE DI UN VETTORE
     ELEMENTATE VALE 4,5,6,7
     RISPETTIVAMENTE PER ALTO,
     DESTRA,BASSO,SINISTRA
```

INDICE DEI I "COME...?"

(i numeri sono quelli di riferimento dei "trucchi")

1 - *Posizionamento del cursore*

2 - *Simulare INPUT X$*

3 - *Come impedire il list di un programma*

4 - *Caricare un programma dopo l'altro*

5 - *Come impedire l'accesso alla tastiera*

6 - *Tutti i comandi sono interpretati come RUN*

7 - *Inibizione del tasto RESET*

8 - *Attesa di un carattere da tastiera*

9 - *Modificare la visualizzazione del list di un programma BASIC*

10 - *GOTO calcolato utilizzando &*

11 - *Stampa con D decimali*

12 - *Incolonnamento a destra in una zona con C caratteri*

13 - *Conoscere l'indirizzo di una variabile*

14 - *Listato su stampante*

15 - *Cambiare la pagina di schermo*

16 - *Modifica della finestra di schermo*

17 - *Per far si che PRINT stampi caratteri minuscoli*

18 - *Cancellazione dello schermo*

19 - *Scrivere sullo schermo dal basso verso l'alto*

20 - *Scrivere sullo schermo da destra a sinistra*

21 - *Spostare tutto il testo verso l'alto*

22 - *Proteggere un INPUT con un valore di default*

23 - *Prevedere la lunghezza di un programma*

24 - *Fare musica con Apple*

25 - *Disegnare, ingrandire far girare una figura*

Pagina zero

Dec	*Hex*	*Nome*	*Funzione*
0,1	00,01	LOC1,LOC2	Utilizzazioni diverse
2-31	02-1F	non utilizzati	disponibili
32,33, 34,35	20,21, 22,23	WNDL,W,T,B	i quattro parametri per la definizione della finestra
36,37	24,25	CH,CV	posizione orizzontale e verticale del cursore
38,39	26,27	GBASL,GBASH	contiene l'indirizzo di base della linea grafica calcolata da GBASCALC dopo ACC
42,43	2A,2B	BAS2L,BAS2H	indirizzo di base relativo alla linea modificata con lo scroll in SCRL1
44,45	2C,2D	H2,V2	parametri di tracciamento orizzontale e verticale in grafica
44,45	2C,2D	LMNEM,RMNEM	codice mnemonico (3 caratteri in 2 byte) per il disassemblatore
46	2E	MASK	0F (linee pari), F0 (dispari) in colore GR
46	2E	CHKSUM	indice di errore di lettura su cassetta con READ
46	2E	FORMAT	per il disassemblaggio degli operandi
47	2F	LASTIN	indicatore di fine lettura su cassetta
47	2F	LENGTH	lunghezza di una istruzione (1, 2 o 3 byte)
48	30	COLOR	indicatore di colore per 2 linee adiacenti

INDIRIZZI DEL MONITOR

Dec	*Hex*	*Nome*	*Funzione*
49	31	MODE	conserva i caratteri dei comandi : . + - o zero
50	32	INVFLG	maschera di inversione dei caratteri da visualizzare
51	33	PROMPT	carattere '*' che avverte la disponibiltà dell'input da tastiera
52	34	YSAV	conserva il registro Y per la NXTITM
53	35	YSAV1	conserva il registro Y prima di COUTZ
54,55	36,37	CSWL,CSWH	contiene l'indirizzo della routine di output che dev'essere usata dal sistema
56,57	38,39	KSWL,KSWH	contiene l'indirizzo della routine di input che dev'essere usata dal sistema
58,59	3A,3B	PCL,PCH	salvataggio del counter, ad esempio prima di un BREAK
60,61	3C,3D	A1L,A1H	memoria di lavoro per MOVE, VFY, READ, WRITE
62,63	3E,3F	A2L,A2H	
64,65	40,41	A3L,A3H	
66,67	42,43	A4L,A4H	
68,69	44,45	A5L,A5H	
69	45	ACC	locazioni di salvataggio dei registri utilizzati da RESTORE e SAVE
70	46	XREG	
71	47	YREG	
72	48	STATUS	
73	49	SPNT	
78,79	4E,4F	RNDL,RNDH	contatore di incremento durante KEYIN. Serve ad innescare la funzione RND
74-77	4A-4D	non utilizzati	disponibili per altri sistemi
80-255	50-FF	non utilizzati	disponibili per altri sistemi

Pagina uno

256-511	100-1FF	Stack	catasta la cui cima è puntata dal registro S

Pagina due

Dec	*Hex*	*Nome*	*Funzione*
512-758	200-2F8	IN	buffer di input riempito da GETLN con ADDIMP

Pagina tre

1016-1018	3F8-3FA	USRADR	JMP all'indirizzo se CTRL-Y
1019-1021	3FB-3FD	NMI	JMP all'indirizzo se interrupt non mascherato come RESET
1022,1023	3FE,3FF	IRQLOC	indirizzo se interruzione hardware

Solo AUTOSTART

1008,1009	3F0,3F1	BRKV	indirizzo di ripresa se BRK è per default 59,FA (OLDBREAK)
1010,1011	3F2,3F3	SOFTEV	indirizzo di ripresa se RESET è per default 03,E0 (BASCONT)
1012	3F4	PWREDUP MASK	byte che decide il boot a freddo o a caldo se premuto RESET

Dec	*Hex*	*Nome*	*Funzione*
1024	400	LINE 1	inizio pagina TEXT o GR
1024-1063	400-427	linea 0 (testo)	linea 0 e 1 in GR
1064-1103	428-44F	linea 8 (testo)	linea 16 e 17 in GR
1104-1143	450-477	linea 16 (testo)	linea 32 e 33 in GR
1144+s	478+s		memoria disponibile ai programmi per l'interfacciamento delle periferiche (1 byte per ogni connettore)
1152-1191	480-4A7	linea 1 TEXT	linee 2 e 3 in GR
1192-1231	4A8-4CF	linea 9 TEXT	linee 18 e 19 in GR
1232-1271	4D0-47F	linea 17 TEXT	linee 34 e 35 in GR
1272+s	4F8+s		memoria RAM utilizzabile per le periferiche connesse in s da 0 a 7
1280-1319	500-527	linea 2 TEXT	linee 4 e 5 in GR
1320-1359	528-54F	linea 10 TEXT	linee 20 e 21 in GR
1360-1399	550-577	linea 18 TEXT	linee 36 e 37 in GR
1400+s	5F8+s		memoria RAM utilizzabile per le periferiche connesse in s
1408-1447	580-5A7	linea 3 TEXT	linee 6 e 7 in GR
1448-1487	5A8-5CF	linea 11 TEXT	linee 12 e 13 in GR
1488-1527	5D0-5F7	linea 19 TEXT	linee 20 e 21 in GR
1528+s	5F8+s		periferiche connesse ad s
1536-1575	600-627	linea 4 TEXT	linee 8 e 9 in GR
1576-1615	628-64F	linea 12 TEXT	linee 24 e 25 in GR
1616-1655	650-677	linea 20 TEXT	linee 40 e 41 in GR
1656+s	678+s		periferiche connesse ad s
1644-1703	680-6A7	linea 5 TEXT	linee 10 e 11 in GR
1704-1743	6A8-6CF	linea 13 TEXT	linee 26 e 27 in GR
1744-1783	6D0-6F7	linea 21 TEXT	linee 42 e 43 in GR

Pagina quattro (seguito)

Dec	*Hex*	*Nome*	*Funzione*
1784+s	6F8+s		periferiche connesse ad s
1792-1831	700-727	linea 6 TEXT	linee 12 e 13 in GR
1832-1871	728-74F	linea 14 TEXT	linee 28 e 29 in GR
1872-1911	750-777	linea 22 TEXT	linee 44 e 45 in GR
1912+s	778+s		periferiche connesse ad s
1920-1959	780-7A7	linea 7 TEXT	linee 14 e 15 in GR
1960-1999	7A8-7CF	linea 15 TEXT	linee 30 e 31 in GR
2000-2039	7D0-7F7	linea 23 TEXT	linee 46 e 47 in GR
2040 2040+s	7F8 7F8+s	SLOT #	contiene $Cs se in s vi è la periferica attualmente utilizzata. Memoria per la periferica connessa in s

Pagina dodici

Dec	*Hex*	*Nome*	*Funzione*
-16384	C000	KBD	indirizzo riservato ai caratteri ricevuti da tastiera. Il bit 7 è posto a 1 se un tasto viene premuto
-16368	C010	KBDSTB	rimette a 0 il bit 7 di C000 prima di un nuovo tasto (POKE)
-16352	C020	TAPEOUT	uscita tutto-niente su cassetta (PEEK)
-16336	C030	SPKR	attivatore tutto-niente dell'altoparlante (PEEK)
-16320	C040	Utility Strobe	invia un impulso sul contatto 5 del connettore della paddle (PEEK)
-16304	C050	TXTCLR	modo grafico (cancellazione) (POKE)
-16303	C051	TXTSET	modo testo (POKE)
-16302	C052	MIXCLR	modo non misto (POKE)
-16301	C053	MIXSET	modo misto (4 linee di testo)

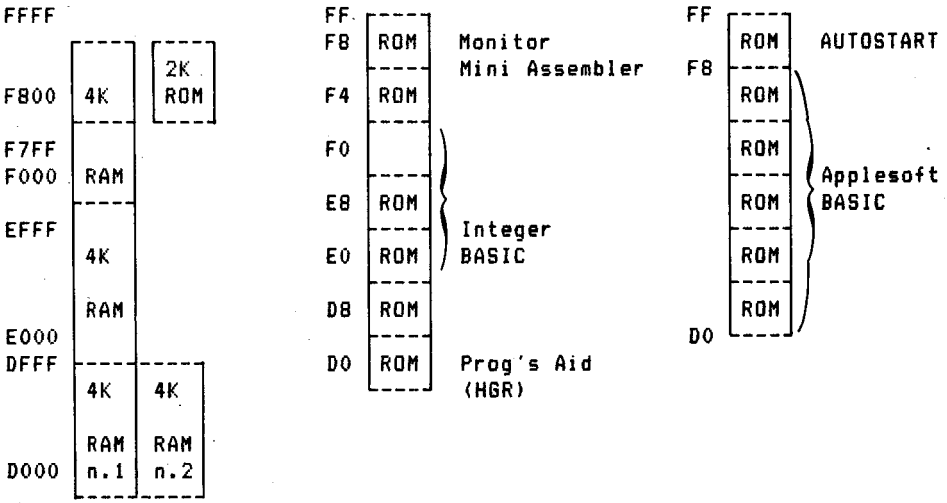
INDIRIZZI DI SISTEMA

Dec	Hex	Nome	Funzione
-16300	C054	LOWSCR	pagina 1 (no cancel.) (POKE)
-16299	C055	HISCR	pagina 2 (no cancel.) (POKE)
-16298	C056	LO-RES	grafica bassa risoluzione (POKE)
-16297	C057	HI-RES	grafica alta risoluzione (POKE)
-16296	C058	SETAN0	SET 0.3v (POKE)
-16295	C059	CLRAN0	CLR 3.5v
-16294	C05A	SETAN1	AN0, AN1, AN2, AN3: 4 linee
-16293			di input logici
-16292	C05C	SETAN2	
-16291	C05D	CLRAN2	
-16290	C05E	SETAN3	
-16289	C05F	CLRAN3	
-16288	C060	TAPIN	lettura del segnale su cassetta memorizzato nel bit 7 (PEEK)
-16287	C061	PB0	
-16286	C062	PB1	PB pulsante paddle, bit 7 a 1
-16285	C063	PB2	se premuto (PEEK)
-16284	C064	PADDL0	
-16283	C065	PADDL1	PADDL paddle per i giochi
-16282	C066	PADDL2	(bit 7) verificato da PREAD
-16281	C067	PADDL3	($FB1E)
-16272	C070	PTRIG	inizializzazione del contatore per la lettura delle paddle (o degli ingressi analogici)

Pagina dodici (seguito)

Dec	*Hex*	*Nome*	*Istruzioni*	*Effetto*
-16256	C080	R	LDA$C080	seleziona la scheda linguaggi **solo in lettura** (protetta in scrittura)
-16255	C081	W ENABLE ROM	LDA$C081 LDA$C081	disattiva la scheda linguaggi e disattiva la protezione in scrittura (**solo scrittura**)
-16254	C082	OFF ENABLE ROM	LDA$C082	disattiva la scheda linguaggi in lettura (protetta in scrittura)
-16253	C083	R/W	LDA$C083 LDA$C083	disativa la scheda linguaggi in scrittura e la seleziona in lettura (**lettura/scrittura**)
-16252, -16247	C084, C087	scheda linguag.	medesimo effetto di C080 - C083	
-16248, -16245	C088, C08B		medesimo effetto di C080 - C087	
-16244, -16241	C08C, C08F	scheda linguag.	questo è il blocco n.1 che è selezionato tra D000 e DFFF quando C080, C087, si tratta del blocco n.2 ENABLE ROM: le ROM della scheda madre e della scheda linguaggi sono attivate in lettura	

riepilogo della mappa di memoria tra D000 e FFFF



scheda linguaggi
16 Kbyte RAM

scheda madre
Integer BASIC

scheda madre
Applesoft BASIC

Pagina dodici ($C0-CFFF)

Dec	*Hex*	*Nome*	*Funzione*
-16240, -16225	C090, C09F	DEV SELECT 1	periferica 1 selezionata con 16 indirizzi riservati per il dialogo
-16234,	C0A0-C0AF	DEV SELECT 2	periferica 2
-16208,	C0B0-C0BF	DEV SELECT 3	periferica 3
-16192,	C0C0-C0CF	DEV SELECT 4	periferica 4
-16176,	C0D0-C0DF	DEV SELECT 5	periferica 5
-16160,	C0E0-C0EF	DEV SELECT 6	periferica 6
-16144	C0F0=C0FF	DEV SELECT 7	periferica 7
	Cs00-CsFF	- inidirizzi di sottoprogrammi in ROM posti in ciascuna scheda di interfaccia per la periferica connessa allo slot s - ciascun sottoprogramma è limitato a 256 byte	
-14336, -12289	C800, CFFF	- estensione dello spazio di memoria ROM eventualmente installata sulla scheda di interfaccia si attiva DEV SELECT s si disattiva con CLRROM	
-12289	CFFF	CLRROM	disattiva l'eventuale espansione ROM $C800 - $CFFF

INDIRIZZI MEMORIA ROM

ROM **Mini-Assemblatore - sweet 16**
Integer + Monitor
Prog's Aid

Indirizzi Dec	Hex	*Funzione*
-2816	F500	Mini-Assemblatore
-2458	F666	Entrate
-2423	F689	Sweet 16 interprete per programmare una pseudo macchina a 16 bit
-2054	F7FA	
-12288	D000	Prog's Aid (grafici ad alta risoluzione)
-11265	D3FF	
-11076	D4BC	Prog's Aid (correzione di programmo Integer)
-10955	D535	Prog's Aid (verifica la registr. su cassetta)
-10531	D6DD	Prog's Aid (Rinumerazione in Integer)
-10473	D717	Prog's Aid (musica)
-8192	E000	Integer
-8189	E003	boot 'a freddo' (CTRL-B)
-2049	F7FF	boot 'a caldo' (CTRL-C)

Comandi		Indirizzi di sottoprogrammi	
Codifica	Nome	Nome	Indirizzi
BC	CTRL-C	BASCONT	FEB3
B2	CTRL-Y	USR	FECA
BE	CTRL-E	REGZ	FEBF
B2	CTRL-Y	USR	FEDA (AUTOSTART)
o ED	T	o TRACE	FEC2 (MONITOR)
EF	V	VFY	FE36
C4	CTRL-K	INPRT	FE8D
B2	CTRL-Y	USR	FEC5 (AUTOSTART)
o EC	S	o STEPZ	FEC4 (MONITOR)
A9	CTRL-P	OUTPRT	FE97
BB	CTRL-B	XBASIC	FEB0
A6	-	SETMODE	FE18
A4	+	SETMODE	FE18
06	M	MOVE	FE2C
95	<	LT	FE20
07	N	SETNORM	FE84
02	I	SETINV	FE80
05	L	LIST	FE5E
F0	W	WRITE	FECD
00	G	GO	FEB6
EB	R	READ	FEFD
93	:	SETMODE	FE18
A7	.	SETMODE	FE18

INDIRIZZI DI MONITOR

Comandi		*Indirizzi di sottoprogrammi*	
Codifica	*Nome*	*Nome*	*Indirizzi*
C6	RETURN	CRMON	FEF6
99	spazio	BLANK	FE04

La codifica dei caratteri di comando è come quella che appare nella tabella dei comandi: CHRTBL $FFCC-$FFE2. La formula di derivazione del codice ASCII a questo codice consiste in 2 operazioni successive:

```
EOR # B0            (vedere NXTCHR in $FFAD)
ADC # 88 (C = 1)
```

La tabella dei vettori relativi ai comandi (indirizzi dei sottoprogrammi) SUBTBL: $FFE3-$FFF9 contiene solo la parte bassa degli indirizzi -1; la parte alta e costante è uguale a $FE.

Le funzioni caratteristiche del MONITOR

1 - Inizio

-155	FF65	MON	ingresso nel monitor con un 'bip'
-151	FF69	MONZ	ingresso nel monitor senza un 'bip'

2 - Input dei dati (posti in $200 su X caratteri)

-665	FD67	GETLINZ	lettura di una linea di comandi
-662	FD6A	GETLN	come GETLINZ con * visualizzato
-651	FD75	NXTCHAR	lettura del prossimo carattere trovato con '→'
-715	FD35	RDCHAR	lettura di un tasto con controllo di ESC
-756	FD0C	RDKEY	lettura di un tasto con cursore lampeggiante
-741	FD1B	KEYIN	carattere nell'Accum. INC di RNDL,H
-721	FD2F	ESC	gestione della posizione del cursore
-707	FD3D	NOTCR	controllo di CTRL-X, BS e 248 caratteri massimi
-636	FD84	ADDINP	aggiunta nel buffer d'ingresso $200 fino a RETURN

3 - Analisi dei dati e interpretazione

-144	FF70	Analisi del buffer $200 con MODE = 0	
-141	FF73	NXTITM	analisi della voce seguente
-89	FFA7	GETNUM	recupero di un numero esadecimale
-83	FFAD	NXTCHR	invio di un carattere e decodifica
-118	FFBA	DIG	ASCII verso A2L,H di un numero
-134	FF7A	CHRSRCH	ricerca del comando
-66	FFBE	TOSUB	chiamata del sottopr. corrispondente
-488	FE18	SETMODE	MODE=':' o '.' o '+' o '-'
-132	FF7C	LZMODE	pone a zero MODE

4 - Visualizzazione dei registri

-550	FDDA	PRBYTE	contenuto di A in 2 cifre esad.
-1724	F944	PRNTX	contenuto di X
-1727	F941	PRNTAX	contenuto di A,X in 4 cifre esad.
-1728	F940	PRNTYX	contenuto di Y,X in 4 cifre esad.
-541	FD3E	PRHEX	peso inferiore di A in 1 cifra esad.
-321	FEBF	REGZ	tutti i registri
-1321	FAD7	RGDSP	sono visualizzati
-1318	FADA	RGDSP1	A, X, Y, P, S

5 - Uscita dei caratteri

-626	FD8E	CROUT	salto alla linea seguente
-531	FDED	COUT	JMP (CSWL)
-528	FDF0	COUT1	visualizzazione di 1 carattere (ACC)
-522	FDF6	COUTZ	con salvataggio di A e Y
-1160	FB78	VIDWAIT	sospensione della visualizzazione con CTRL-S
-1027	FBFD	VIDOUT	invia i tipi di caratteri visualiz-
-1040	FBF0	STORADV	zati e posiziona il cursore
-198	FF3A	BELL	emissione di un 'bip'
-1063	FBD9	BELL1	ritarda un 'bip'
-1052	FBE4	BELL2	attiva un 'bip'
-1720	F948	PRBLNK	3 spazi
-384	FE80	SETINV	visualizzazione in modo inverso
-380	FE84	SETNORM	visualizzazione in modo normale

6 - Gestione del cursore

-1036	FBF4	ADVANCE	sposta di una posizione a destra
-926	FC62	CR	all'inizio della linea seguente
-922	FC66	LF	alla linea seguente
-1008	FC10	BS	una posizione a sinistra
-998	FC1A	UP	alla linea precedente
-990	FC22	VTAB	alla linea specificata in Acc CV
-1189	FB5B	TABV	alla linea specificata in Acc CV
-1087	FBC1	BASCALC	calcola l'indirizzo di base BASL, BASH secondo A

7 - Gestione del testo

-1223	FB39	SETTXT	modo testo
-1205	FB4B	SETWND	finestra di testo
-936	FC58	HOME	cancellazione nella finestra di testo
-958	FC42	CLREOP	cancellazione fino alla parte bassa dello schermo
-868	FC9C	CLREOL	cancellazione fino alla fine della linea
-912	FC70	SCROLL	scroll verso l'alto

8 - Grafici in bassa risoluzione

-1216	FB40	SETGR	modo GR misto. Cancellazione
-1948	F864	SETCOL	colore specificato in A COLOR
-2048	F800	PLOT	
-2034	F80E	PLOT1	vedere dettagli nella pagina seguente
-2023	F819	HLINE	
-2020	F81C	HLINE1	
-2010	F826	VLINEZ	
-2008	F828	VLINE	
-1998	F832	CLRSCR	
-1994	F836	CLRTOP	
-1997	F847	GBASCALC	calcola l'indirizzo base della linea grafica specificata in A
-1935	F871	SCRN	pone in A il colore del quadratino posto in Y, A

Comandi MONITOR

Chiamata da BASIC	*MONITOR*	*Nome*	*Risultato*
CALL-2048	F800G	PLOT	A, Y — calcolo di GBASL,H
CALL-2034	F80EG	PLOT1	Y — all'ordinata corrente
CALL-2023	F819G	HLINE	A, Y ($2C) — A e Y sono modificati
CALL-2020	F81CG	HLINE1	Y ($2C) — alla linea corrente
CALL-2010	F826G	VLINEZ	Y, (A)+1+(C) ($2D) A viene modificato
CALL-2008	F828G	VLINE	Y, A ($2D)
CALL-1998	F832G	CLRSCR	@ — cancellazione dello schermo grafico che viene posto in modo testo @
CALL-1994	F836G	CLRTOP	cancellazione dello schermo grafico. Le 4 linee in basso non vengono modificate
CALL-1992	F838G	CLRSC2	Y cancellazione fino alla linea Y
CALL-1988	F83CG	CLRSC3	Y, ($2D) cancellazione dello schermo nella parte in alto a sinistra

INDIRIZZI DI MONITOR

9 - Input/Output

-1425	FA6F	INITAN	inizializzazione delle uscite logiche ($C058-$C05F)
-1250	FB1E	PREAD	lettura della paddle, n. in X, risultato in Y
-371	FE8D	INPRT	'IN#s' per altri input
-375	FE89	SETKBD	pone in linea la tastiera
-361	FE97	OUTPRT	'PR#s' per altri output
-365	FE93	SETVID	pone in linea il video (PR=0)
-307	FECD	WRITE	scrive su cassetta da (A1) fino ad (A2)
-259	FEFD	READ	lettura e memorizzazione tra (A1) e (A2)
-823	FCC9	HEADR	scrittura su cassetta
	FAB4	verifica il controller del dischetto	

10 - Disassemblaggio

-418	FE5E	LIST	'L', disassembla 20 istruzioni
-413	FE63	LIST2	disassemblaggio di (A) istruzioni
-1840	F8D0	INSTDSP	una istruzione
-1918	F882	INSDS1	il suo indirizzo
-1906	F88C	INSDS2	il suo codice operativo
-1879	F8A9	GETTFMT	ricerca il tipo di istruzione
-1709	F953	PCADJ	aggiorna il counter

11 - Visualizzazione delle locazioni di memoria

-589	FDB3	XAM	visualizza da (A1) a (A2)
-622	FD92	PRA1	visualizza l'indirizzo seguito da -
-586	FDB6	DATAOUT	visualizza i contenuti
-838	FCBA	NXTA1	incremento di (A1) fino a (A2) (C=1)

12 - Spostamento e verifica del contenuto di memoria

-468	FE2C	MOVE	spostamento di (A1)-(A2) verso (A4)
-844	FCB4	NXTA4	incremento di A4L,H
-458	FE36	VFY	verifica di (A1)-(A2) con (A4)
-480	FE20	LT	trasferisce da A2 in A4 e A5

13 - Aritmetica esadecimale

-570	FDC6	XAMPM	addizione o sottrazione di A1L e A2L con '+' o '-' nell'accumulatore. Il risultato viene visualizzato

14 - Supervisore

-1472	FA40	IRQ	interrupt hardware
-1447	FA59	OLDBREAK	"BRK" in linguaggio macchina
-1438	FA62	RESET	RESET
-1407	FA81	NEWMON	invia il comando di inizializzazione
-1370	FAA6	PWRUP	inizializzazione 'a freddo'

-1367	FAA9	SETPG3	sistemazione dei vettori $3F0 e $3F3
-1169	FB6F	SETPWRC	calcola il valore da porre in $3F4
-856	FCA8	WAIT	temporizzazione
-336	FEB0	XBASIC	inizializzazione del BASIC
-333	FEB3	BASCONT	si procede in BASIC
-330	FEB6	GO	'G'
-266	FEF6	CRMON	solo se premuto RETURN
-310	FECA	USR	JMP $3F8 se premuto CTRL-Y
-193	FF3F	RESTORE	ripristino di A, X, Y, P, S
-182	FF4A	SAVE	salvataggio di A, X, Y, P, S

Differenze

AUTOSTART		*MONITOR*	
STEP	non esiste	FA40-FA85 FAA5-FAD6 FAFD-FB18	- disassembla le istruzioni in corso, modo passo-passo
IRQ	FA40	FA86	- considera una interruzione
BREAK	FA4C	FA92	o un arresto
OLDBREAK	FA59	XBRK FA9C	- mostra il PD e i registri
RESET	FA62-FAA3	non esiste	- auto-boot
APPLE II	FB60	non esiste	- mostra APPLE II all'accensione
SETPWRC	FB6F	non esiste	- memorizzazione di PWREDUP
VIDWAIT	FB78	non esiste	- interrompe e riprende la visualizzazione con CTRL-S
NOWAIT	FB94	non esiste	
ESCOLD	FB97	non esiste	- posizionamento del cursore con ESC I, J, K, M
ESCNOW	FB9B	non esiste	
ESCNEW	FBA5	non esiste	
MULPM	non esiste	FB60-FB80	- moltiplicazione intera 16 bit
DIVPM	non esiste	FB81-FBC0	- divisione intera
TRACE	non esiste	FEC2	- modo TRACE (vedere STEP)
STEPZ	FEC4	FEC4	- in AUTOSTART conduce a USR

INDIRIZZI FONDAMENTALI

SOFTEV e PWREDUP $3F2, $3F3 e $3F4

La ROM AUTOSTART, il MONITOR in ROM disponibile sull'Apple II determinano il tipo di programma in caso di RESET.
Il RESET 'a caldo' è programmabile nel vettore SOFTEV e il MONITOR trattiene un byte particolare PWREDUP come traccia del passaggio per un rilancio 'a freddo'. PWREDUP dev'essere un exclusive OR $$A5 e del contenuto di $3F3 se si desidera che RESET produca un rilancio 'a caldo'.

Valore di default:

	SOFTEV	PWREDUP
senza DOS	$E003	$45 (BASIC 'a caldo')
senza DOS	$9DBF	$38

Se si inizializza il sistema 'a freddo', accendendolo, viene riconosciuta la presenza della scheda di interfaccia per le unità floppy disk, in questo caso il Monitor permette l'esecuzione del programma che carica il DOS (sistema operativo), (Bootstrap) presente nella ROM della scheda di interfaccia.

Nell'insieme delle funzioni di 'messa in marcia' del DOS, vi è l'assegnazione di un indirizzo per il rilancio 'a caldo' (in caso di RESET) che viene realizzato in SOFTEV ($9DBF) (parte bassa in testa).

SOFTEV può essere modificato dall'utente che avrà cura di assegnare a PWREDUP l'exclusive OR di SOFTEV+1 e di $A5 per far si che RESET porti ad un programma specificato e non provochi l'equivalente di un rilancio 'a freddo'.

AMPERV $3F5, $3F6, $3F7

L'indirizzo AMPERV sarà utilizzato (indirettamente) per dare inizio all'esecuzione di un programma in linguaggio macchina da un programma in Applesoft che contiene la parola-chiave '&'.

Esempio: * 3F5 : 4C 00 03 JMP $300

La parola-chiave '&' porterà all'esecuzione del sottoprogramma che comincia all'indirizzo $0300.

DOSWARMSTART $3D0 : 4C BF 9D

Questo indirizzo è utilizzato per tornare all'Applesoft (controllo da DOS) dopo l'utilizzo del Monitor, battere:

```
* 3D0G
]
```

Il programma corrente non è stato toccato.

MONZ $FF69 (-151)

Indirizzo di ingresso al Monitor.
Dall'Applesoft, si batte
CALL -151
poi viene visualizzato *

Routine fondamentali

Stampa di un carattere

COUT	$FDED	Visualizza il carattere presente nell'Accumulatore e sposta il cursore di una posizione. Considera RETURN, LF e i 2 modi Normal/Inverse.
OUTDO	$DB5C	In Applesoft, visualizza il carattere presente nel l'accumulatore e tiene in considerazione i 3 modi di visualizzazione Normal/Inverse/Flash.

Acquisizione di un carattere in un testo BASIC

CHRGET	$00B1	Questo sottoprogramma (che si automodifica) punta in $B8, $B9 il carattere da prendere che sarà cambiato nell'Accumulatore Z=1 se fine dell'istruzione ($3A o $00) C=0 se il carattere è una cifra (gli spazi del testo in BASIC sono saltati).
CHRGOT	$00B7	Il carattere è il carattere attuale e non il seguente come in CHRGET.

Lettura di un carattere battuto in tastiera

RDKEY	$FD0C	Attende che un tasto sia premuto con il cursore lampeggiante. Il codice del carattere è caricato nell'Accumulatore.

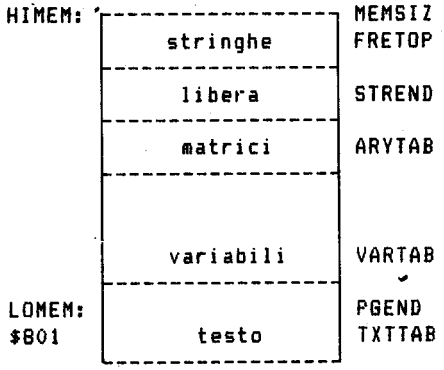
PUNTATORI FONDAMENTALI DELL'APPLESOFT
(indirizzi in ordine crescente)

Nome	*Indirizzo*		*Funzione*
	Hex	*Dec*	
TXTTAB	$67,$68	103,104	Inizio del testo BASIC = $801 (2049) per default
VARTAB	$69,$6A	105,106	Inizio delle variabili semplici ,dei puntatori delle stringhe, dei puntatori delle funzioni.
ARYTAB	$6B,$6C	107,108	Inizio delle variabili dimensionate, puntatori alle matrici di di stringhe.
STREND	$6D,$6E	109,110	Inizio spazio libero
FRETOP	$6F,$70	111,112	Fine spazio libero. Fine delle stringhe.
MEMSIZ	$73,$74	115,116	Inizio stringhe. Fine spazio in memoria +1. Le stringhe sono registrate dall'alto verso il basso.
CURLIN	$75,$76	117,118	Numero della linea in corso di esecuzione.
OLDLIN	$77,$78	119,120	N. di linea interrotta da CTRL-C , STOP o END.
OLDTXT	$79,7A	121,122	Indirizzo dell'ultimo byte (00) della linea in corso di esecuzione.
DATLIN	$7B,$7C	123,124	N. della linea nella quale sono letti i DATA.
DATPTR	$7D,$7E	125,126	Indirizzo del primo byte relativo ai DATA da leggere.
INPPTR	$7F,$80	127,128	Puntatore del buffer d'ingresso da tastiera durante INPUT.
VARNAM	$81,$82	129,130	Contiene il nome (2 caratteri) dell'ultima variabile a cui si è fatto riferimento.

PUNTATORI FONDAMENTALI DELL'APPLESOFT
(indirizzi in ordine crescente)

Nome	*Indirizzo*		*Funzione*
	Hex	*Dec*	
VARPNT	$83,$84	131,132	Indirizzo del valore dell'ultima variabile a cui si è fatto riferimento, o del byte relativo alla lunghezza di una stringa.
PGEND	$AF,$B0	175,176	Fino del testo BASIC.

Riassunto



INDIRIZZI

APPLESOFT - ESEMPIO N.1

Implementazione di un programma e delle variabili in RAM

```
 *
  LIST

10 AA = 2
20 AA% = 4
30 AA$ = ""
40  DIM AA(1,3)
50  DIM AA%(2,1)
60  DIM AA$(3,2)
70  DEF  FN AA(X) = X - 256 *INT
      (X / 256)
80  PRINT  FN AA(257)


 RUN
1

 CALL-151

*69.6A

0069- 74 08
*800.873

0800- 00 0A 08 0A 00 41 41 D0
0808- 32 00 14 08 14 00 41 41
0810- 25 D0 34 00 1F 08 1E 00
0818- 41 41 24 D0 22 22 00 2C
0820- 08 28 00 86 41 41 28 31
0828- 2C 33 29 00 3A 08 32 00
0830- 86 41 41 25 28 32 2C 31
0838- 29 00 48 08 3C 00 86 41
0840- 41 24 28 33 2C 32 29 00
0848- 63 08 46 00 B8 C2 41 41
0850- 28 58 29 D0 58 C9 32 35
0858- 36 CA D3 28 58 CB 32 35
0860- 36 29 00 71 08 50 00 BA
0868- C2 41 41 28 32 35 37 29
0870- 00 00 00 0A
```

AA variabile reale semplice
AA% variabile semplice intera
AA$ variabile semplice stringa di caratteri
AA(1,3) variabile reale dimensionata
AA%(2,1) variabile intera dimensionata
AA$ variabile stringa dimensionata)
FN AA(X) funzione definita dal programma

Testo codificato in BASIC

<> indirizzo di inizio della linea di istruzione seguente

[]numero della linea di istruzione

(parola riservata

00 fine linea

00 00 fine del programma

---- definizione di funzione

Variabili semplici

```
*6B.6C
                                  2 caratteri del nome ';'
006B- 97 08                       valore '.'
*874.896

0874-.41 41;82 00
0878- 00 00 00.C1 C1;00 04 00
0880- 00 00.41 C1;00,1D 08 00
0888- 00·C1 41;54 08,92 08,58.
0890-.58 00;00 00 00 00 00.
```

Codifica dei due caratteri del nome

		1° car.	2° car.
AA	reale	P	P
AA%	intera	N	N
AA$	stringa	P	N
AA(X)	funzione	N	P

P è il codice ASCII con bit 7=0
N è il codice ASCII con bit 7=1

Valore di una variabile su 5 byte

	1°byte	2°byte	3°byte	4°byte	5°byte
Reale	esponente	mantissa			
Intera	parte bassa	parte alta	non utilizzati		
Stringa	lunghezza	indirizzo di inizio stringa		non utilizzati	
Funzione	indirizzo della definizione		indirizzo dell'argomento		codice P del 1° carattere dopo =

Gli indirizzi sono espressi con la parte bassa sul primo byte e la parte alta sul secondo byte.

Implementazione di variabili dimensionate

```
*6D.6E

006D- 0A 09
*897.909

0897-.41                                  DIM AA(1,3)
0898- 41;<31 00> 02 00 04 00 02
08A0-/00 00 00 00 00/00 00 00
08A8- 00 00/00 00 00 00 00/00
08B0- 00 00 00 00/00 00 00 00
08B8- 00/00 00 00 00 00/00 00
08C0- 00 00 00/00 00 00 00 00
08C8-.C1 C1;<15 00> 02 00 02 00           DIM AA%(2,1)
08D0- 03/00 00/00 00/00 00/00
08D8- 00/00 00/00 00/41 C1;<2D            DIM AA$(3,2)
08E0- 00> 02 00 03 00 04 00 00
08E8- 00/00 00 00/00 00 00/00
08F0- 00 00/00 00 00/00 00 00/
08F8- 00 00 00/00 00 00/00 00
0900- 00/00 00 00/00 00 00/00
0908- 00 00/
```

'.' 2 caratteri per il nome ; offset / variabile seguente

numero delle dimensioni o degli indici

numero massimo degli elementi della matrice per ciascuna dimensione dall'ultima alla prima (valore massimo dell'indice + 1).

// valore di ciascun elemento della matrice
- reale 5 byte
- intera 2 byte soltanto
- stringa 3 byte: lunghezza, indirizzo

nell'ordine AA(0,0), AA(1,0), AA(0,1), AA(1,1), AA(0,2), AA(1,2), AA(0,3), AA(1,3) l'indice più a destra aumenta più lentamente.

Esempio di implementazione di un programma e delle variabili in memoria RAM

```
LIST

10  REM  APPLESOFT
20  INPUT "NOME ?";N$
30  FOR K = 1 TO  LEN (N$) - 2
40  PRINT  RIGHT$ (N$, LEN (N$) -
     K + 1);" "; MID$ (N*,K,2);"
     ";
50  NEXT K

CALL-151

*69.6A                                  VARTAB

0069- 65 08

*67.68

0067- 01                                TXTTAB
0068- 08

*800.862

0800- 00 11 08 0A 00 B2 20 41
0808- 50 50 4C 45 53 4F 46 54
0810- 00 22 08 14 00 B4 22 4E
0818- 4F 4D 45 20 3F 22 3B 4E
0820- 24 00 33 08 1E 00 81 4B
0828- D0 31 C1 E3 28 4E 24 29
0830- C9 32 00 5B 08 28 00 BA
0838- E9 28 4E 24 2C E3 28 4E
0840- 24 29 C9 4B C8 31 29 3B
0848- 22 20 22 3B EA 28 4E 24
0850- 2C 4B 2C 32 29 3B 22 20
0858- 22 3B 00 62 08 32 00 82
0860- 4B 00 00
```

Testo BASIC codificato

(parole riservate

00 fine della linea

00 00 fine del programma

I I n° di linea

<> indirizzo della linea seguente

APPLESOFT - ESEMPIO N.2

Esempio di implementazione di un programma e delle variabili in memoria RAM
(segue)

```
 RUN
NOME ?SERGIO
SERGIO SE ERGIO ER RGIO RG GIO GI
```

esecuzione delle istruzioni

```
 CALL-151

*6D.6E
```

STREND

```
006D- 73 08

*863.870

0863-.00 0A;4E,80 06/
0868- FA 95.00 00;4B 00 83 20
0870- 00/
```

variabili

-N$: lunghezza, puntatore
-K : valore reale

```
*6F.70

006F- FA
0070- 95
```

FRETOT

```
*95FA.95FF

95FA- 53 45 52 47 49 4F
```

Valore di N$

```
*79.7A

0079- 61 08
```

indirizzo dell'ultimo byte dell'ultima istruzione eseguita

```
*AF.B0

00AF- 65
00B0- 08
```

fine del programma (+2)

Le parole riservate in ordine crescente per codice e gli indirizzi dei sottoprogrammi corrispondenti nell'interprete.

Codici da $80 a $A9

Parola riservata	*Codice decimale*	*Indirizzo decimale*
END	$80 128	$D870 55408
FOR	$81 129	$D766 55142
NEXT	$82 130	$DCF9 56569
DATA	$83 131	$D995 55701
INPUT	$84 132	$DBB2 56242
DEL	$85 133	$F331 62257
DIM	$86 134	$DFD9 57305
READ	$87 135	$DBE2 56290
GR	$88 136	$F390 62352
TEXT	$89 137	$F399 62361
PR=	$8A 138	$F1E5 61925
IN=	$8B 139	$F1DE 61918
CALL	$8C 140	$F1D5 61909
PLOT	$8D 141	$F225 61989
HLIN	$8E 142	$F232 62002
VLIN	$8F 143	$F241 62017
HGR2	$90 144	$F3D8 62424
HGR	$91 145	$F3E2 62434
HCOLOR=	$92 146	$F6E9 63209
HPLOT	$93 147	$F6FE 63230
DRAW	$94 148	$F769 63337
XDRAW	$95 149	$F76F 63343
HTAB	$96 150	$F7E7 63463
HOME	$97 151	$FC58 64600
ROT=	$98 152	$F721 63265
SCALE=	$99 153	$F727 63271
SHLOAD	$9A 154	$F775 63349
TRACE	$9B 155	$F26D 62061
NOTRACE	$9C 156	$F26F 62063
NORMAL	$9D 157	$F273 62067
INVERSE	$9E 158	$F277 62071
FLASH	$9F 159	$F280 62080
COLOR=	$A0 160	$F24F 62031
POP	$A1 161	$D96B 55659
VTAB	$A2 162	$F256 62038
HIMEM:	$A3 163	$F286 62086
LOMEM:	$A4 164	$F2A6 62118
ONERR	$A5 165	$F2CB 62155
RESUME	$A6 166	$F318 62232
RECALL	$A7 167	$F3BC 62396
STORE	$A8 168	$F39F 62367
SPEED=	$A9 169	$F262 62050

Le parole riservate in ordine crescente per codice e gli indirizzi dei sottoprogrammi corrispondenti nell'interprete.

Codici da $AA a $D1

Parola riservata	*Codice decimale*	*Indirizzo decimale*
LET	$AA 170	$DA46 55878
GOTO	$AB 171	$D9E3 55614
RUN	$AC 172	$D912 55570
IF	$AD 173	$D9C9 55753
RESTORE	$AE 174	$D849 55369
&	$AF 175	$03F5 1013
GOSUB	$B0 176	$D921 55585
RETURN	$B1 177	$D96B 55659
REM	$B2 178	$D9DC 55772
STOP	$B3 179	$D86E 55406
ON	$B4 180	$D9EC 55788
WAIT	$B5 181	$E784 59268
LOAD	$B6 182	$D8C9 55497
SAVE	$B7 183	$D8B0 55472
DEF	$B8 184	$E313 58131
POKE	$B9 185	$E77B 59259
PRINT	$BA 186	$DAD5 56021
CONT	$BB 187	$D896 55446
LIST	$BC 188	$D6A5 54949
CLEAR	$BD 189	$D66A 54890
GET	$BE 190	$DBA0 56224
NEW	$BF 191	$D649 54857
TAB(	$C0 192	
TO	$C1 193	
FN	$C2 194	
SPC(	$C3 195	
THEN	$C4 196	
AT	$C5 197	
NOT	$C6 198	$DE98 -8552
STEP	$C7 199	
+	$C8 200	$E7BE -6210
-	$C9 201	$E7A7 -6233
*	$CA 202	$E97F -5761
/	$CB 203	$EA66 -5530
^	$CC 204	$EF09 -4349
AND	$CD 205	$DF55 -8363
OR	$CE 206	$DF4F -8369
>	$CF 207	
=	$D0 208	$DF6A -8342
<	$D1 209	

Le parole riservate in ordine crescente per codice e gli indirizzi dei sottoprogrammi corrispondenti nell'interprete.

Codici da $D2 a $EA

Parola riservata	*Codice decimale*	*Indirizzo decimale*
SGN	$D2 210	$EB90 60304
INT	$D3 211	$EC23 60451
ABS	$D4 212	$EBAF 60335
USR	$D5 213	$000A 10
FRE	$D6 214	$E2DE 58078
SCRN(	$D7 215	$D412 54290
PDL	$D8 216	$DFCD 57293
POS	$D9 217	$E2FF 58111
SQR	$DA 218	$EE8D 61069
RND	$DB 219	$EFAE 61358
LOG	$DC 220	$E941 59713
EXP	$DD 221	$EF09 61193
COS	$DE 222	$EFEA 61418
SIN	$DF 223	$EFF1 61425
TAN	$E0 224	$F03A 61498
ATN	$E1 225	$F09E 61598
PEEK	$E2 226	$E764 59236
LEN	$E3 227	$E6D6 59094
STR$	$E4 228	$E3C5 58309
VAL	$E5 229	$E707 59143
ASC	$E6 230	$E6E5 59109
CHR$	$E7 231	$E646 58950
LEFT$	$E8 232	$E65A 58970
RIGHT$	$E9 233	$E6B6 59014
MID$	$EA 234	$E691 59025

INDIRIZZI DELL'INTERPRETE APPLESOFT
Classificati per grandi funzioni

1 - Entry point (punti di entrata)

-8192	E000	BASIC	inizializzazione 'a freddo' se CTRL-B
-3800	F128	COLDSTART	inizializzazione 'a freddo' se CTRL-B
-8189	E003	BASIC2	2° ingresso detto 'a caldo' se CTRL-C o RESET
-11204	D43C	CMDLOOP	lancio del loop principale dell'interprete (a caldo)

2 - Input dei dati (nel buffer $200)

-11201	D43F	input con la visualizzazione di]	
-10962	D52E	INLIN+2	input da tastiera di una istruzione con (X) come 'prompt'
-10964	D52C	INLIN	input senza 'prompt'
-10951	D539	GDBUFS	pone a zero il bit 7 di tutti i caratteri registrati da INLIN
-10925	D553	INCHR	input di un carettere nell'accumulatore e messa a zero del bit 7

3 - Analisi dei dati (nella zona riservata al programma)

177 183	B1 B7	CHRGET CHRGOT	caricamento nell'accumulatore del carattere puntato da ($B8, $B9) o TXTPTR (seguente o attuale) e discriminazione del tipo C=0 per le cifre Z=1 per la fine di una linea o di una istruzione
-9716	DA0C	LINGET	caricamento in $50, $51 o LINNUM del numero della linea puntata da $B8, $B9 o TXTPTR e prosecuzione dell'analisi
-6411	E6F5	GTBYTC	acquisizione di un carattere con CHRGET e valutazione a partire da TXTPTR per X
-6408	E6F8	GETBYT	valutazione dell'espressione puntata da TXTPTR e trasferimento del risultato in
-6405	E6FB	CONINT	in FAC poi FAC→intero≤255 in X e FACLO
-10726	D61A	FNDLIN	ricerca nel programma l'indirizzo dell'istruzione il cui numero è in LINNUM ($50, $51). Se C=1, il risultato in LOWTR ($9B, $9C) altrimenti LOWTR punta all'istruzione di numero più elevato
-6234	E74C	COMBYTE	verifica che TXTPTR punti su una virgola e continui l'analisi con GETBYT
-6330	E746	GETNUM	acquisizione di un numero per la valutazione e test della virgola sul carattere seguente

(segue)

-8857	DD67	FRMNUM	valuta una espressione puntata da TXPTR e pone il risultato in FAC assicurandosi che sia un numero
-6318	E752	GETADR	FAC→intero (2 byte) in $50, $51
-2087	F7D9	GETARYPT	ricerca di una variabile con il nome puntato da TXTPTR e ...
-8221	DFE3	PTRGET	... il risultato in $83, $84 o VARPNT e Y, A - se non esiste già creazione - il posto del nome nella tavola delle variabili è in $9B, $9C o LOWTR
-8067	E07D	ISLETC	l'Accumulatore contiene un codice ASCII di una lettera, si C = 1
-8526	DEB2	PARCHK	verifica delle parentesi
-8520	DEB8	CHKCLS	TXTPTR punta su)?
-8517	DEBB	CHKOPN	TXTPTR punta su (?
-8514	DEBE	CHKCOM	TXTPTR punta su ,?
-8512	DEC0	SYNCHR	altrimenti, errore di sintassi se si l'analisi continua
	EC4A	FIN	registra il numero flottante puntato da TXTPTR in FAC

4 - Visualizzazione dei dati

-4818	ED2E	PRNTFAC	visualizza FAC ($9D-A2) e lo distrugge
-9414	DB3A	STROUT	visualizza la stringa puntata da (Y, A)
-9411	DB3D	STRPRT	visualizza la stringa puntata da (FACMO, FACLO)
-9385	DB57	OUTSP	visualizza uno spazio
-9477	DAFB	CRDO	ritorno carrello
-9382	DB5A	OUTQST	?
-9380	DB5C	OUTDO	visualizza l'Accum. con i modi I, F, N
-4839	ED19	INPRT	visualizza "IN" n° di linea corrente
-4828	ED24	LINPRT	visualizza un intero in X, A
-9515	DAD5	PRINT	istruzione di stampa su video

5 - Aritmetica e funzioni algebriche

COSTANTI NUMERICHE

E0FE	90	80	00	00	20	-2^{16} = -32767.0005
ED0A	9B	3E	BC	1F	FD	99 999 999,9
EDF0	9E	6E	6B	27	FD	999 999 999
ED14	9E	6E	6B	28	00	1 000 000 000 = 10^9

INDIRIZZI DELL'INTERPRETE APPLESOFT
(segue)

F066	81	49	0F	DA	A2	PI/2 = 1.57079633
F06B	83	49	0F	DA	02	2xPI = 6.28318531
F070	7F	00	00	00	00	1/4
EE64	80	00	00	00	00	1/2
E913	81	00	00	00	00	1
E92D	80	35	04	F3	34	SQR(0 5) = 0.707106781
E932	81	35	04	F3	34	SQR(2) = 1.41421356
E937	80	80	00	00	00	-1/2
E93C	80	31	72	17	F8	LOG(2) = .693147181
EA50	84	20	00	00	00	10
EEDB	81	38	AA	3B	2A	LOG(e)/LOG(2) = 1.44269504

FUNZIONI

E7A0	FADDH	(FAC) ← (FAC) + 1/2
E7A7	FSUB	ARG ← (Y,A) e chiama FSUBT
E7AA	FSUBT	FAC ← ARG-FAC
E7BE	FADD	ARG ← (Y,A) e chiama FADDT
E7C1	FADDT	FAC ← FAC + ARG
E941	LOG	FAC ← LN(FAC)
E97F	FMULT	ARG ← (Y,A) e chiama FMULTT
E982	FMULTT	ARG ← FAC * ARG
E9E3	CONUPK	ARG ← (Y,A)
EA39	MUL10	FAC ← FAC * 10
EA55	DIV10	FAC ← FAC / 10
EA66	FDIV	ARG ← (Y(A)) e chiama FDIVT
EA69	FDIVT	FAC ← ARG / FAC
EB80	SGN	FAC ← segno di FAC
EB82	SIGN	A ← segni di FAC (1 se > 0, 0 se 0, FF se < 0)
EB93	FLOAT	FAC ← A diventa flottante
EBAF	ABS	valore assoluto FAC ← FAC
EC23	INT	il più grande valore intero inferiore a FAC ← FAC
EBF2	QINT	il più grande valore intero inferiore se FAC < 32767
E10C	AYINT	il più grande valore intero inferiore nella mantissa FAC

6 - Funzioni sulle stringhe di caratteri

DEC	*HEX*		
-8837	DD7B	FRMEVL	valutazione di una espressione partendo da TXTPTR
-8575	DE81	STRTXT	TXTPTR → Y,X, segue chiamata a STRLIT
-7193	E3E7	STRLIT	mette un carattere alla fine della stringa in ENDCHR

-7187	E3ED	STRLT2	costruisce un descrittore di stringa in DSCTMP, FACMO, LO e conduce a PUTNEW
-7126	E42A	PUTNEW	pone DSCTMP in un descrittore temporaneo puntato da FACMO, LO
-7203	E3DD	STRSPA	conduce a GETSPA e pone il puntatore e la lunghezza in DSCTMP
-7086	E452	GETSPA	libera dello spazio per una stringa spostando verso il basso FRESPC e FRETOP - può emettere "OUT OF MEMORY" - aggiorna DSCTMP
-6761	E597	CAT	concatenazione della stringa descritta da (FACMO, LO) e quella puntata da TXTPTR +1
-6686	E5E2	MOVSTR	sposta la stringa puntata da Y, X con lunghezza A nella posizione puntata da FRESPC ($71, $72)
-6700	E5D4	MOVINS	sposta la stringa il cui descrittore è puntato da STRNF1 verso FRESPC
-6659	E5FD	FRESTR	verifica che FAC indirizzi una stringa e conduce a FRAFAC
-6656	E600	FREFAC	libera lo spazio occupato per una stringa temporanea
-6603	E635	FRETMS	libera il descrittore temporaneo senza liberare la stringa
-7036	E484	GARBAG	recupera lo spazio occupato dalle stringhe abbandonate spostando verso l'alto le altre

VARIABILI UTILIZZATE PAGINA ZERO

DEC	*HEX*		
13	D	CHRAC = "	
14	E	END CHR = 00	
17	11	VALTYP = 1	se stringa in FAC
82	52	TEMPPT	puntatore temporaneo
83	53	LASTPT	puntatore temporaneo
	5E,5F	INDEX	puntatore temporaneo

INDIRIZZI DELL'INTERPRETE APPLESOFT

DEC	*HEX*		
111,112	6F,70	FRETOP	zona bassa stringhe
113,114	71,72	FRESPC	fine della zona libera
133,134	85,86	FORPNT	utilizzata da COPY per liberare spazio
148,149	94,95	HIGHDS	utilizzato per BLTU
150,151	96,97	HIGHTR	per l'indirizzo di destinazione
155,156	9B,9C	LOWTR	utilizzato per BLTU
157,158,159	9D,9E,9F	DSCTMP	descrittore di stringa
160,161	A0,A1	FACMO,LO	puntatore di descrittore
171,172	AB,AC	STRNG1	puntatore utilizzato da MOVINS
173,174	AD,AE	STRNG2	puntatore utilizzato da STRLT2

7 - Funzioni grafiche alta risoluzione

INDIRIZZI

pagina 1 : $2000 - $3FF7
pagina 2 : $4000 - $5FF7

linea 0 : $2000 - $2027 (40 byte / linea)
linea 64 : $2028 - $204F
linea 128 : $2050 - $2077
linea 8 : $2080 - $20A7
linea 72 : $20A8 - $20CF
linea 136 : $20D0 - $20F7

linee 16,80?144,24,88,152 : $2100 - $21F7
linee 32,96,160,40,104,168 : $2200 - $22F7
linee 48,112,176,56,120,184 : $2300 - $23F7

Sia n il numero della linea e $a il suo indirizzo le linee precedentemente listate quindi:

la n+1 linea ha per indirizzo $a + $400
la n+2 linea ha per indirizzo $a + $800
la n+3 linea ha per indirizzo $a + $C00
la n+4 linea ha per indirizzo $a + $1000
la n+5 linea ha per indirizzo $a + $1400
la n+6 linea ha per indirizzo $a + $1800
la n+7 linea ha per indirizzo $a + $1C00

VARIABILI DELLA PAGINA ZERO

DEC	*HEX*	*NOME*	*FUNZIONE*
26,27	$1A,$1B	SHAPE L,H	puntatore nella tabella delle figure
28	$1C	HCOLOR1	dipende dalla parità dell'ascissa X di HMASK e HCOLOR0
29	$1D	COUNT H	contatore nella traccia di linea
38,39	$26,$27	HBASL,H	indirizzo d'inizio di una linea
48	$30	HMASK	$81, $82, $84, $88 b0, b1, b2, b3 $90, $A0, $C0 b4, b5, b6
82	$52	DY	incremento di y per HLINE
83	$53	QDRNT	angolo di rotazione per DRAW

VARIABILI DELLA PAGINA ZERO UTILIZZATE PER LE FUNZIONI GRAFICHE

DEC	*HEX*	*NOME*	*FUNZIONE*
224,225 226	$E0,$E1 $E2	xH,xL y	coordinate dei punti tracciati da HPLOT
228	$E4	HCOLOR0	00, 2A, 55, 7F nero, ocra, blu, bianco 80, AA, D5, FF nero, verde, rosso, bianco
229	$E5	XD7	n° di byte in una linea per il punto di ascissa x
230	$E6	HPAG	$20: pagina 1 $40: pagina 2
231	$E7	SCALE	fattore di scala di una figura
232,233	$E8,$E9	SHPTAB	puntatore d'inizio della tavola delle figure
234	$EA	CC	contatore dsi collisione

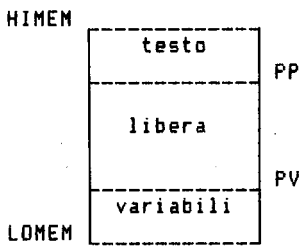
INDIRIZZI DELL'INTERPRETE APPLESOFT

FUNZIONI GRAFICHE

DEC	*HEX*	*NOME*	*FUNZIONE, RISULTATO*
-3112 -3102	F3D8 F3E2	HGR2 HGR	cancellazione delle pagine $20 → MPAG, $40 → HPAG
-3084	F3F4	BKGND	schermo di un colore uniforme colore → Acc → HCOLOR1
	F6F0	HCOLOR	colore → X → HCOLOR0
-3055	F411	HPOSN	x → y,x → H xL y → Acc → y calcolo di HBASL,H; HMASK e XD7 e HCOLOR1
-2971	FA65	INTX	seguito da N, incremento o decremento di xH, xL e y
-2861	F4D3	INTY	seguito da N, incremento o decremento di y → HBASL,H
-2613	F5CB	IPOS	HBASL,H; XD7 → xH, xL e y
-2985	F457	HPLOT	tracciamento del punto x → y, x e y → A con l'aiuto
-2982	F4A5	PLOT	tracciamento del punto definito da HCOLOR1; HMASK; XD7 → Y; HBASL,H seguendo le istruzioni: LDA HCOLOR1 EOR (HBASL),Y AND HMASK EOR (HBASL),Y STA (HBASL),Y
-2758	F53A	HLINE	tracciamento di una linea in qualunque direzione dal punto attuale al punto x → X,A e y → Y

Nome	*Indirizzi*		*Funzioni*
	Hex	*Dec*	
LOMEM	$4A,$4B	74,75	Inizio delle variabili
HIMEM	$4C,$4D	76,77	Fine del testo BASIC
PP	$CA,$CB	202,203	Inizio del testo BASIC
PV	$CC,$CD	204,205	Fine delle variabili
PR	$DC,$DD	220,221	N° della linea corrente
PN	$DE,$DF	222,223	Nome dell'ultima variabile a cui si è fatto riferimento
PX	$E0,$E1	224,225	Inizio dell'istruzione corrente

Riepilogo



INTEGER - ESEMPIO

Esempio di implementazione di un programma e delle sue variabili in memoria RAM. (Vedere esempio n° 2 in Applesoft).

```
>LIST
   10 REM  INTEGER
   15 DIM N$(20)
   20 INPUT "NOME ?",N$
   30 FOR K=1 TO LEN(N$)-2
   40 PRINT N$(K);" ";N$(K,K+1);" "
      ;
   50 NEXT K
   60 END

>CALL-151

*4A.4B                                    LOMEM

004A- 00 08

*CA.CB

00CA- 9A 95                               PP


*4C.4D

004C- 00 96                               HIMEM

*959A.95FF


959A- (0D)[0A 00](5D A0 C9                Testo BASIC
95A0- CE D4 C5 C7 C5 D2 01 (0C)
95A8- [0F 00](4E CE (40 (22 B2 14         ( parole riservate
95B0- 00 (72 01 (10)[14 00](53 (28
95B8- CE CF CD C5 A0 BF (29 (26           01 fine linea
95C0- CE (40 01 (13)[1E 00](55 CB
95C8- (56 B1 01 00 57 3B CE 40            <> numero di byte
95D0- 72 13 B2 02 00 01 (1F)[28           della linea -1
95D8- 00](61 CE (40 (2A CB (72 (45
95E0- (28 A0 (29 (45 CE (40 (2A CB        I I n° di linea
95E8- (23 CB (12 B1 01 00 (72 (45
95F0- (28 A0 (29 (47 01 (06)[32 00]
95F8- 59 CB 01 (05)[3C 00](51 01

*CC.CD

00CC- 00 08                               PV
```

Esempio di implementazione di un programma e delle sue variabili in memoria RAM.
(segue)

```
>RUN
NOME ?SERGIO

SERGIO SE ERGIO ER RGIO RG GIO GI
```

Esecuzione del programma

```
>CALL-151

*CC.CD

00CC- 20 08

*800.81F

0800-.CE 40 00 <1A 08>/D3 C5 D2
0808- C7 C9 CF/1E 53 4F 46 54
0810- 00 22 08 14 00 84 22 4E
0818- 4F 4D.CB 00<20 08>/05 00/
```

PV

Variabili

- N$,<puntatore alla variabile successiva>
- valore (20 caratteri)
- K,<>,valore

INDIRIZZI INTEGER

Le parole riservate per ordine crescente di codice e gli indirizzi dei sottoprogrammi corrispondenti nell'interprete Integer BASIC. Codici da $00 a $15.

Parola riservata	*Codice*		*Indirizzo*	
	Hex	*Dec*	*Hex*	*Dec -65536*
[HIMEM:]	$00	0		
Fine linea	$01	1		
-	$02	2		
:	$03	3		
LOAD	$04	4	$F0DF	-3873
SAVE	$05	5	$F140	-3776
CON	$06	6	$F30A	-3318
RUN (n° di linea)	$07	7	$EFF2	-4110
RUN	$08	8	$EFEC	-4116
DEL	$09	9	$E36F	-7313
, (per DEL)	$0A	10		
NEW	$0B	11	$E6AD	-6739
CLR	$0C	12	$E5B7	-6729
AUTO	$0D	13	$E7E2	-6174
, (per AUTO)	$0E	14		
MAN	$0F	15	$EE54	-4524
HIMEM:	$10	16	$F04D	-4019
LOMEM:	$11	17	$F0C9	-3895
+	$12	18	$E787	-6267
-	$13	19	$E782	-6270
*	$14	20	$E222	-7646
/	$15	21	$EF10	-4336

Le parole riservate (segue). Codici da $16 a $2C

Parola riservata		Codice Hex	Codice Dec	Indirizzo Hex	Indirizzo Dec -65536
=	operatori logici per i numeri	$16	22		
#		$17	23		
> =		$18	24		
>		$19	25		
< =		$1A	26		
< >		$1B	27		
<		$1C	28		
AND		$1D	29		
OR		$1E	30		
MOD		$1F	31	$E27A	-7558
^		$20	32	$F371	-3215
		$21	33		
((per DIM)		$22	34		
(per DIM)		$23	35		
THEN (n° di linea)		$24	36		
THEN (istruz.)		$25	37		
, (stringa)		$26	38		
, (numero)		$27	39		
" (inizio)		$28	40		
" (fine)		$29	41		
(var.$		$2A	42		
indice		$2B	43		
		$2C	44		

Le parole riservate (segue). Codici da $2D a $40.

Parola riservata	*Codice*		*Indirizzo*	
	Hex	*Dec*	*Hex*	*Dec -65536*
(var. indice	$2D	45		
PEEK	$2E	46	$EEF6	-4362
RND	$2F	47	$EF4E	-4274
SGN	$30	48	$E75C	-6308
ABS	$31	49	$E74A	-6326
PDL	$32	50	$F33B	-3269
	$33	51		
(per DIM	$34	52		
+ (segno)	$35	53		
- (segno)	$36	54		
NOT	$37	55	$E736	-6346
(	$38	56		
= comparazione	$39	57		
# di stringhe	$3A	58		
LEN(	$3B	59	$EE22	-4574
ASC(	$3C	60	$F31D	-3299
SCRN(	$3D	61	$E28A	-7542
, in SCRN	$3E	62		
(	$3F	63		
$ (stringa)	$40	64		

Le parole riservate (segue). Codici da $41 a $56.

Parola riservata	*Codice* Hex	*Codice* Dec	*Indirizzo* Hex	*Indirizzo* Dec -65536
	$41	65		
(	$42	66		
,	$43	67		
,	$44	68		
;	$45	69		
;	$46	70		
;	$47	71		
,	$48	72		
,	$49	73		
,	$4A	74		
TEXT	$4B	75		
GR	$4C	76		
CALL	$4D	77	$EEA0	-4448
DIM (stringhe)	$4E	78	$E130	-7888
DIM (numeri)	$4F	79	$EF1E	-4322
TAB	$50	80	$E7A4	-6236
END	$51	81		
INPUT (stringa)	$52	82	$E171	-7823
INPUT (messaggio)	$53	83		
INPUT (numero)	$54	84	$EBAA	-5206
FOR	$55	85	$E93A	-5830
= (FOR/NEXT)	$56	86		

Le parole riservate (segue). Codici da $57 a $6B.

Parola riservata	*Codice*		*Indirizzo*	
	Hex	*Dec*	*Hex*	*Dec -65536*
TO (STEP)	$57	87	$E950	-5808
STEP	$58	88	$F279	-3463
NEXT	$59	89		
, NEXT	$5A	90		
RETURN	$5B	91	$E8A5	-5979
GOSUB	$5C	92	$E83C	-6084
REM	$5D	93		
LET	$5E	94		
GOTO	$5F	95	$E85B	-6053
IF	$60	96	$E828	-6104
PRINT (stringa)	$61	97	$EE03	-4605
PRINT (numero)	$62	98		
PRINT	$63	99		
POKE	$64	100		
, (POKE)	$65	101		
COLOR=	$66	102	$EE4E	-4530
PLOT	$67	103	$EE3F	-4545
, (PLOT)	$68	104		
HLIN	$69	105	$EEB0	-4432
, (HLIN)	$6A	106		
AT (HLIN)	$6B	107		

Le parole riservate (segue). Codici da $6C a $7F.

Parola riservata	*Codice* Hex	*Codice* Dec	*Indirizzo* Hex	*Indirizzo* Dec -65536
VLIN	$6C	108	$EEC6	-4410
, (VLIN)	$6D	109		
AT (VLIN)	$6E	110		
VTAB	$6F	111	$EE57	-4521
= (stringa)	$70	112		
= (numero)	$71	113		
)	$72	114		
	$73	115		
LIST	$74	116		
, (LIST)	$75	117		
LIST	$76	118		
POP	$77	119	$F167	-3737
NODSP (stringa)	$78	120		
NODSP (numero)	$79	121		
NOTRACE	$7A	122	$F176	-3722
DSP (stringa)	$7B	123	$F2E0	-3360
DSP	$7C	124		
TRACE	$7D	125	$F171	-3727
PR#	$7E	126	$F3C9	-3127
IN#	$7F	127	$F41A	-3046

DOS: INDIRIZZI DISCHETTI

BOOT: utilizzo del DOS (inizializzazione a freddo)

Programma	*Localizzazione*	*Occupazione*	*Funzione*
1 - BOOT 0	PROM scheda controllo $C600	256 byte	carica BOOT 1 in RAM
2 - BOOT 1	DISCHETTO: traccia 0, settore 0 RAM: $800-$900	1 settore 256 byte	carica BOOT 2 e se stesso
3 - BOOT 2	DISCHETTO: traccia 0,s.da 1 a 9 MASTER SLAVE (48K) $3700-$4000 $B700-$C000	9 settori 2304 byte	contiene RWTS carica il DOS ed eventual- mente il rilo- catore
BOOT 1	DISCHETTO: traccia 0, settore 0 MASTER SLAVE (48K) $3600-$36FF $B600-$B6FF	1 settore 256 byte	versione del BOOT 1 dispo- nibile per l'inizializza- zione di un disco vergine
4 - DOS	DISCHETTO: traccia 2, set.da 4 a 0 traccia 1, set.da F a 0 traccia 0, set.da F a C MASTER SLAVE (48K) $1D00-$3600 $9D00-$B600	 25 settori 6400 byte	sistema opera- tivo dei co- mandi e ges- tione dello spazio sul dischetto
Rilocatore	DISCHETTO MASTER traccia 0, set. A e B RAM: $1B00-$1D00 (non esiste sul dis- chetto SLAVE)	 2 settori 512 byte	reinstallazio- ne del DOS al suo posto de- finitivo $9D00-$C000 (48 Kbyte)

Organizzazione del dischetto

Qualunque sia la versione del DOS, un dischetto è costituito da 35 tracce e i dati sono divisi in settori da 256 byte.

Vesione 3.2	*Versione 3.3*
13 settori/traccia 455 settori/dischetto di cui 403 utili quindi 103168 byte utili	16 settori/traccia 560 settori/dischetto di cui 496 utili quindi 126976 byte utili

Occupazione di tracce e settori

Traccia 0, 1, 2	: DOS	(sistema operativo)
Traccia $11, settore 0	: VTOC	(occupazione)
Traccia $11, settori da $F a $1	: DIRECTORY	(catalogo)
Tracce da $12 a $22 e	: programmi e archivi	
da $16 a $3	: utilizzatore	

(L'archivio più lungo, registrabile su un dischetto può essere costituito da circa 126000 byte).

La "directory" può contenere il nome di 105 riferimenti.

Un riferimento è un insieme di 35 caratteri che comprendono:
- l'indirizzo dell'elenco dei settori occupati (n° della traccia, n° del settore) dal file referenziato
- il tipo di file A, I, T, B, e la sua eventuale protezione contro la cancellazione accidentale
- il nome del file (30 caratteri)
- la lunghezza e il numero dei settori occupati (2 byte)

Comandi in ordine di apparizione nella tabella dei comandi

Comandi DOS index			*Indirizzo di entrata Hex*	*DEC*
$00	0	INIT	$A54F	42319
$01	1	LOAD	$A413	42003
$02	2	SAVE	$A397	41879
$03	3	RUN	$A4D1	42193
$04	4	CHAIN	$A4F0	42224
$05	5	DELETE	$A263	41571
$06	6	LOCK	$A271	41585
$07	7	UNLOCK	$A275	41589
$08	8	CLOSE	$A2EA	41706
$09	9	READ	$A51B	42267
$0A	10	EXEC	$A5C6	42438
$0B	11	WRITE	$A510	42256
$0C	12	POSITION	$A5DD	42461
$0D	13	OPEN	$A2A3	41635
$0E	14	APPEND	$A298	41624
$0F	15	RENAME	$A281	41601
$10	16	CATALOG	$A56E	42350
$11	17	MON	$A233	41523
$12	18	NOMON	$A23D	41533
$13	19	PR#	$A229	41513
$14	20	IN#	$A22E	41518
$15	21	MAXFILES	$A251	41553
$16	22	FP	$A57A	42362
$17	23	INT	$A59E	42398
$18	24	BSAVE	$A331	41777
$19	25	BLOAD	$A35D	41821
$1A	26	BRUN	$A38E	41870
$1B	27	VERIFY	$A27D	41597

Posizione	*DOS*	*Punto entrata*
$B600-B6FF	RWTS	$B7B5
$AAC9-B5FF	gestione dei comandi	$AAFD
$9D00-AAC8	programma principale	$9D00
$9600-9CFF	3 buffer da 595 byte	

Configurazione da 48 Kbyte di memoria RAM

RWTS (Read - Write - Track - Sector)
(lettura - scrittura - traccia - settore)

Sottoprogrammi di accesso ad un settore: RWTS

Tabella dei parametri: IOB

L'indirizzo di IOB è caricato nel registro A (parte alta) e in Y (parte bassa) prima della chiamata di RWTS:

```
Esempio: LDA =$10
         LDY =$00
         JSR $3D9
         RTS

* 1000 : 01 60 01 00 11 0C 11 10
* 1008 : 00 09 00 00 01
* 1011 : 00 01 EF D8
```

```
IOB: byte n° 4 : n° di traccia ($11)
     byte n° 5 : n° di settore ($0C)
     byte 6, 7 : indirizzo di DCT ($1011)
     byte 8, 9 : indirizzo della zona di
                 trasferimento in RAM
     byte C    : codice del comando
                 00 posizionamento
                 01 lettura
                 02 scrittura
                 03 formattazione
```

DCT costanti : 00 01 EF D8 della periferica

Indirizzo Hex	*Contenuto*	*Funzione*
3D0	JMP $9DBF	inizializzazione 'a caldo'
3D3	JMP $9D84	inizializzazione 'a freddo'
3D6	JMP $AADF	gestione dei comandi
3D9	JMP $B7B5	lettura/scrittura di un settore (RWTS)
3DC	LDA $9D0F LDY $9D0E RTS	ricerca dell'indirizzo relativo al'e- lenco dei parametri per la gestione dei comandi
3E3	LDA $AAC2 LDY $AAC1 RTS	ricerca dell'indirizzo relativo alla tabella IOB dei parametri dell'RWTS
3EA	JMP $A851	per sostituire i vettori di I/O $38, $39 e $36, $37 con i puntatori del DOS
3EF	JMP $FA59	in caso di BRK in MONITOR
3F3,3F2	$9DBF	indirizzo di rinvio in caso di RESET (SOFTEV)
3F4	$38	PWRUP = ($3F3) + $A5
3F5	JMP $FF58	in caso di &
3F8	JMP $FF65	in caso di CTRL-Y
3FB	JMP $FF65	se interrupt non mascherato
3FE	$FF65	se interrupt

Indirizzi diversi del DOS in memoria RAM

- File binari caricati in RAM con BLOAD
  indirizzo in $AA72, AA73
  lunghezza in $AA60, AA61

- Programma eseguito con l'inizializzazione 'a freddo'
  nome: traccia 1, settore 9, byte $75...
  type * 9E42 : 34 BRUN
  * 9E42 : 14 EXEC

- Cessazione della pausa attivata durante CATALOG
  AE34 : 60

SYSTEM MASTER 3.3

B	FID	- copia di file o programmi - verifica tasso di occupazione sul disco
A I	COPY	- copia integrale di un disco (con uno o due unità per floppy disk)
B	BOOT13	- inizializzazione del sistema con il DOS versione 3.2 (13 settori per traccia)
B	MUFFIN	- conversione di file o programmi scritti sotto DOS 3.2 in file e programmi sotto DOS 3.3
B	MASTER CREATE	- creazione di un disco MASTER partendo da un disco già inizializzato e utilizzabile come disco SLAVE - possibilità di modificare il nome di un programma che dev'essere eseguito al BOOTSTAP

DAIKIN5 programming aid's 3.3

The Patcher	visualizzazione e modifica di un qualunque settore
The Peeker	lettura di un file

Implementazione di file e di programmi sul dischetto

esempio:

1 - estratto del CATALOG

```
DISK VOLUME 254

*A 006 HELLO
*I 018 ANIMALS
*T 003 APPLE PROMS
*I 006 APPLESOFT
*I 026 APPLEVISION
*I 017 BIORHYTHM
*B 010 BOOT13
```

* protezione in scrittura
A,I,T,B tipi di file
n: numero dei settori occupati

2 - estratto della DIRECTORY del dischetto

```
          Traccia $11        Settore $0F

00   00 <11 0E> 00 00 00 00 00
08   00 00 00 [13 0F] (82)/C8 C5          HE
10   CC CC CF A0 A0 A0 A0 A0   LLO
18   A0 A0 A0 A0 A0 A0 A0 A0
20   A0 A0 A0 A0 A0 A0 A0 A0
28   A0 A0 A0 A0/[06 00] [14 0F]
30   (81)/C1 CE C9 CD C1 CC D3   ANIMALS
38   A0 A0 A0 A0 A0 A0 A0 A0
40   A0 A0 A0 A0 A0 A0 A0 A0
48   A0 A0 A0 A0 A0 A0 A0/[12
50   00 [15 0F] (80)/C1 D0 D0 CC     APPL
58   C5 A0 D0 D2 CF CD D3 A0   E PROMS
60   A0 A0 A0 A0 A0 A0 A0 A0
68   A0 A0 A0 A0 A0 A0 A0 A0
70   A0 A0/[03 00] [16 0F] (81)/C1        A
78   D0 D0 CC C5 D3 CF C6 D4   PPLESOFT

80   A0 A0 A0 A0 A0 A0 A0 A0
88   A0 A0 A0 A0 A0 A0 A0 A0
90   A0 A0 A0 A0 A0/[06 00] [17
98   0F] (81)/C1 D0 D0 CC C5 D6   APPLEV
A0   C9 D3 C9 CF CE A0 A0 A0   ISION
A8   A0 A0 A0 A0 A0 A0 A0 A0
B0   A0 A0 A0 A0 A0 A0 A0 A0/
B8   [1A 00] [18 0F] (81)/C2 C9 CF     BIO
C0   D2 C8 D9 D4 C8 CD A0 A0   RHYTHM
C8   A0 A0 A0 A0 A0 A0 A0 A0
D0   A0 A0 A0 A0 A0 A0 A0 A0
D8   A0 A0 A0/[11 00] [19 0F] (84)
E0  /C2 CF CF D4 B1 B3 A0 A0   BOOT13
E8   A0 A0 A0 A0 A0 A0 A0 A0
F0   A0 A0 A0 A0 A0 A0 A0 A0
F8   A0 A0 A0 A0 A0 A0 [0A 00]
```

<> traccia, settore del seguente file

[ ] traccia, settore dell'elenco dei settori occupati da questo file

( ) tipo di file
82 Applesoft
81 Integer
80 file T
84 binario

// nome del file completato per 30 caratteri con degli spazi

└ lunghezza del file e numero dei settori

DOS - ESEMPIO (segue)

3 - estratto della VTOC o tavola d'occupazione dei settori

Traccia $11 Settore 0

```
00  04 11 0F 03 00 00 FE 00
08  00 00 00 00 00 00 00 00
10  00 00 00 00 00 00 00 00
18  00 00 00 00 00 00 00 00
20  00 00 00 00 00 00 00 7A
28  00 00 00 00 00 00 00 00
30  0D FF 00 00 23 10 00 01
38  00 00 00 00 00 00 00 00
40  00 00 00 00 00 00 00 00
48  00 00 00 00 00 00 00 00
50  00 00 00 00 00 00 00 00
58  00 00 00 00 00 0F 00 00
60  FF FF 00 00 00 00 00 00
68  00 7F 00 00 01 FF 00 00
70  00 00 00 00 00 00 00 00
78  00 00 00 00 00 00 00 00

80  FF E0 00 00 00 00 00 00
88  00 00 00 00 00 00 00 00
90  00 00 00 00 00 00 00 00
98  00 00 00 00 00 00 00 00
A0  00 00 00 00 00 03 00 00
A8  00 00 00 00 00 00 00 00
B0  00 00 00 00 00 00 00 00
B8  00 00 00 00 00 00 00 00
C0  00 00 00 00 00 00 00 00
```

[ ] traccia/settore del 1° settore del contenuto

V versione del DOS
Λ (3.3) n° del volume (254)

= 122 settori max. nell'elenco degli indirizzi (traccia/settore) dei settori occupati da un file

= 35 tracce max. per dischetto

10 = 16 settori per traccia

= 256 byte per settore

settori occupati in ciascuna traccia successiva

FEDCBA98	76543210	

se un settore è libero il corrispondente bit è posto a 1

FF FF 0000 la traccia n° 10 ha tutti i suoi settori liberi

00 7F 0000 la traccia n° 12 ha i settori F,E,D,C,A,9,8,7 occupati

4 - estratto di un elenco di indirizzi (traccia/settore) di settori occupati da un file

Esempio: HELLO, traccia $13, settore $F

```
00- 00 00 00 00 00 00 00 00
08- 00 00 00 00 13 0E 13 0D
10- 13 0C 13 0B 13 0A 00 00
18- 00 00 00 00 00 00 00 00
```

indirizzi dei settori successivamente occupati da HELLO

5 - estratto della coda di un programma registrato su disco

Esempio: HELLO, scritto in Applesoft

traccia $13 settore $E

```
00  71 04 19 08 0A 00 B2 20   Q      2
08  20 2D 2D 20 44 4F 53 20    -- DOS
10  33 2E 33 20 48 45 4C 4C   3.3 HELL
18  4F 00 20 08 14 00 B2 20   O      2
20  00 28 08 1E 00 89 3A BA    (     : :
28  00 2E 08 28 00 97 00 59    .  (    Y
30  08 32 00 BA 22 44 4F 53    2 :"DOS
38  20 56 45 52 53 49 4F 4E    VERSION
40  20 33 2E 33 20 20 20 20    3.3
48  20 20 20 20 20 20 20 20
50  30 38 2F 32 35 2F 38 30   08/25/80
58  22 00 8B 08 3C 00 BA 3A   "    <  : :
60  BA 22 41 50 50 4C 45 20   :"APPLE
68  49 49 20 50 4C 55 53 20   II PLUS
70  4F 52 20 52 4F 4D 43 41   OR ROMCA
78  52 44 20 20 20 53 59 53   RD    SYS
```

[pb pA] lunghezza del programma in numero di byte occupati in RAM

pb: parte bassa
pA: parte alta

/ / istruzioni di programma

DOS - ESEMPIO (segue)

6 - estratto della coda di un programma binario

Esempio: BOOT13, traccia $19, settore $E

```
00  [00 17] [F0 08] 20 E3 03 84
08   00 85 01 A0 01 B1 00 8D
10   90 17 C8 B1 00 8D 91 17
18   20 58 FC A0 FF C8 B9 96
20   17 08 09 80 20 ED FD 28
28   10 F3 A9 BF 85 33 20 6A
30   FD AD 00 02 C9 8D F0 0F
38   C9 B1 90 DC C9 B8 B0 D8
40   0A 0A 0A 0A 8D 82 17 A9
48   17 A0 81 20 00 1D B0 F7
50   AD FE 16 8D 8A 17 85 13
58   E6 13 AD FF 16 4A 4A 4A
60   85 10 A9 17 A0 81 20 00
68   1D B0 F7 EE 8A 17 EE 86
70   17 AD 86 17 C5 10 F0 EA
78   90 E8 AD 82 17 AA A9 00
```

[pb pA] lunghezza del programma in numero di byte in RAM

[ ] indirizzo di RAM di inizio del programma scritto in linguaggio macchina

' istruzioni di programma in linguaggio macchina

BOOT13 è stato salvato sul dischetto con il comando
BSAVE BOOT13, A$8F0, L$1700

Implementazione di un file su dischetto

7 - estratto di un file di tipo T

Esempio: APPLE PROMS, traccia $15, settore $E

```
00  B7 B5 8D C4 C5 CC A0 B1   75 DEL 1
08  B0 B0 B0 AC B1 B2 B5 B0   000,1250
10  8D D3 C1 D6 C5 A0 D2 C1    SAVE RA
18  CE C4 CF CD 8D C8 CF CD   NDOM HOM
20  C5 8D D2 D5 CE 8D 00 00   E RUN
28  D0 C1 D2 C1 CC CC C5 CC   PARALLEL
30  A0 D0 D2 C9 CE D4 AC B2    PRINT,2
38  B5 B6 AC B8 AC B5 B0 B0   56,8,500
40  8D 00 00 00 00 00 00 00
48  00 00 00 00 00 00 00 00
50  C3 CF CD CD D5 CE C9 C3   COMMUNIC
58  C1 D4 C9 CF CE D3 AC B2   ATIONS,2
60  B5 B6 AC B8 AC B1 B2 B5   56,8,125
68  B0 8D 00 00 00 00 00 00   0
70  00 00 00 00 00 00 00 00
78  A8 CE CF D4 A0 C1 D6 C1   (NOT AVA
```

(8D separatore (RETURN)

AC) separatore ","

record di lunghezza fissa

00 nessun dato scritto (END OF DATA)

Questo file è stato aperto con il comando

```
OPEN APPLE PROMS, L40
```

definendo di 40 caratteri la lungheza di ciascun record.

Partendo dal record n° 1 l'istruzione di lettura dei campi è:

```
INPUT N$, BL, BW, ST
```

e di scrittura

```
PRINT N$;",";BL;",";BW;",";ST
```

8 - Estratto del contenuto di un file sequenziale di tipo T

```
00- B3 8D D0 D2 C5 CD C9 C5 3. PRIMO
08- D2 8D D3 C5 C3 CF CE C4 R.SECOND
10- 8D D4 D2 CF C9 D3 C9 C5 .TERZO
18- CD C5 8D 00 00 00 00 00 ........
20- 00 00 00 00 00 00 00 00 ........
```

(8D il separatore <u>RETURN</u> chiude ogni record che ha lunghezza libera.

00 nessun dato scritto (END OF DATA)

L. 15.000

Cod. 340H ISBN 88-7056-140-2

Questo libro è destinato a risiedere, in permanenza, a fianco del vostro Apple. Esso vi ricorda tutte le informazioni relative a riferimenti che potranno essere reperite velocemente: sintassi dei comandi, codici dei caratteri, messaggi di errore, linguaggio macchina, connessioni e indirizzi utili.

Le informazioni sono date senza eccessivi dettagli poiché lo scopo principale di questo libro è quello di fornire un rapido accesso alle informazioni; per una esposizione più introduttiva e completa, si potrà far riferimento ai libri "La pratica dell'Apple II" e "Apple II - Guida all'uso".

Si termina con una raccolta di "trucchi" di differenti livelli, ma tutti utili, i "come...?" sono disposti senza un ordine particolare, è comunque possibile il loro rapido reperimento grazie ad un apposito indice.